iApp

User Manual

Version 4.0

Copyright **InSync Software, Inc.**

Table of Contents

i. Section 1 – Installing iApp

1 Release Notes 6
1.1 Introduction 7
1.2 What's New 7
1.3 Known Issues 8
2 Installation Manual 9
2.1 Introduction 10
2.2 Minimum Hardware and Software Requirements 10
2.3 iApp Software Support 10
2.4 Pre-Requisites 11
2.5 Installing RFID iApp™ 12
2.6 Post Installation Steps 41
2.7 Serial Port Communication 43
2.8 Starting RFID iApp™ 43
2.9 View Log 45
3 Installation Manual - Install Steps on Linux 46
3.1 Introduction 47
3.2 Minimum Hardware and Software Requirements 47
3.3 Pre-Requisites 47
3.4 Pre-Installation Tasks 47
3.5 Setting the profile for Environment Variables 48
3.6 Installing RFID iApp™ Fedora Linux 13 distribution 48
4 iApp Configuration 54
4.1 Introduction 55
4.2 Logging In 55
4.3 Configuration Panel Description 57
4.4 Configuration 60
4.5 Devices 79
4.6 Tag Schemes 84
4.7 Edge Processes 88
4.8 External Device Types 102
4.9 External Devices 105
4.10 Asset Placement 108
4.11 Printers 109
4.12 Printer Documents 111
4.13 Event Management Service 128
4.14 Subscribers 139
4.15 Data Schemes 150
4.16 Formatters 154
4.17 Schedule Jobs 159
4.18 Monitoring 165
4.19 Alerts 170
4.20 Activities 178
4.21 GEN2 181

4.22 File Based Simulation 189
4.23 Post Event 190
4.24 RTLS 191
4.25 Software 196
5 Device Types 208
5.1 Introduction 209
5.2 Device Types 209
6 Edge Process 234
6.1 Introduction 235
6.2 Edge Process Fundamentals 235
6.3 Device Action Groups 237
6.4 Event Providers (EP) 264
6.5 Example of Building Edge Process 268
6.6 Creating Control Object 268
6.7 Creating Event Provider 269
7 iApp Event Management 274
7.1 Introduction 275
7.2 Handler Properties 275
7.3 Event Properties 275
7.4 Handler Properties Section 276
7.5 Event Handlers Available 276
7.6 Create PML Data Handler 277
7.7 Alert Event Handler 278
7.8 Inbound Shared Event Handler 279
7.9 Outbound Shared Event Handler 280
7.10 Job Trigger Handler 283
7.11 Item Tracking Handler 284
7.12 Item History Tracking Handler 286
7.13 Post Event Handler 288
7.14 Inventory Handler 290
7.15 Auto Print Handler 291
7.16 Button Handler 292
7.17 Email Handler 294
7.18 RTLS Alert Event Handler 296
7.19 RTLS Handler 298
7.20 Introduction to Event Handler Design 299
7.21 Naming Convention 299
7.22 Event Handler 299
7.23 Event Action 301
7.24 APIs 302
7.25 How to Create an Event Handler 303
7.26 List of ValueMethods 307
8 iApp UI Services 312
8.1 Introduction 313
8.2 Infrastructure 313
8.3 ServiceName 314
8.4 Action 314

8.5 Parameters 314
9 Image Map Calibration 321
9.1 Image Calibration 322
9.2 Calibration Process: 323
9.3 Image Regions 326
9.4 Locator Tags 334
9.5 Asset Icons 336
10 Flex Image Map Component 337
10.1 Introduction 338
10.2 Components 338
10.3 Image Map Viewer Wrapper 339
10.4 Image Map Viewer 341
10.5 Image Maps Util 343
11 Custom DataGrid Capabilities 345
11.1 Custom DataGrid 346
12 Load Balancing 352
12.1 Introduction 353
12.2 Minimum Hardware and Software Requirements 353
12.3 Load Balancer - Pre-Requisites 353
12.4 Configure Apache for Load Balancing 354
13 iApp Administration Manual 356
13.1 Introduction 357
13.2 Administration Set up 357
13.3 Activities 357
13.4 Instance Registry 358
13.5 Create Sensornet Instance 358
13.6 Create Process Zone Instance 361
13.7 Create Print Server Instance 364
13.8 Organization / Users 365
13.9 Roles and Permissions 368
13.10 Users 372
13.11 Catalogs 374
13.12 _OrgType 375
13.13 _EMSType 376
13.14 _Menu 377
13.15 _PersonType 378
13.16 Add Catalog 378
13.17 Export / Import 381
13.18 Messages 383
13.19 Performance Tuning 387
13.20 User Locks 387
13.21 Organization Locks 388
14 iApp Connection Objects 390
14.1 Introduction 391
14.2 Connection Types 391
14.3 Example: 391
15 iApp Custom Jobs 395

15.1 Introduction 396
15.2 Structure 396
15.3 Example 397
16 Locking Service 402
16.1 Locking Service Overview 403
16.2 Data model 404
16.3 LockingService 404
16.4 Locking UI 405
17 iApp Internationalization 407
17.1 iApp Internationalization Overview 408
18 SOAP Framework 412
18.1 Introduction 413
18.2 How to invoke the SOAP method remotely 413
19 iApp Web Services 417
19.1 About iApp Web Services 418
19.2 Introduction 418
19.3 Creating Material Type/Material/Items 418
19.4 Posting Tag Read and RTLS Tag read event 443
19.5 URL Generation for tag plotting on image 451

i. Section One
Installing iApp

1 Release Notes

1.1 Introduction

This release note documents the features in RFID IApp v4. This document also has the list of known issues in the release.

1.2 What's New

The following are the enhancements.

- Flex 3.0 compliant user interface with easy navigable menus
- Ability to monitor data by enterprise organization for event processing
    - Event notifications
    - Job Status
- Integrated iApp with Identec ILR System, and UBISense, Mojix RTLS Systems
- Ability to install and configure ProcessZone on running SensorNet without a need for restart.
- Receiver Framework at SensorNet to support Multi-Modal Sensor Data Collection
- Ability to switch on/off monitoring to improve performance
- Real Time Location System
    - Support for Geo-Fence (entry/exit) and time-fence (min dwell/max dwell) alerts
    - RTLS image viewer to view the assets in real time with material filters.
    - Mobile regions support based upon material.
    - Ability to drill down on the image viewer.
    - Support for multiple RTLS sites
    - Multi panel search report
    - Density Graph
- Ability to view assets on the image viewer with passive readers to indicate number of assets present near a reader
- Installation procedure without the need for any database client installation on the server.

1.3 Known Issues

Following are the known issues for this release:

Sl. No.	Issue
1	If the Organization name is modified, user needs to logout and relogin to the system to view the changes. Otherwise error messages are displayed and none of the screens work.
2	Connection name change does not reflect in the Event Handler association screen.
3	Action script error occurs when user without any permission, tries to login to sensornet.
4	When the RTLS site doesn't have any image associated with it, action script error is raised on Asset Placement screen
5	User is able to create zone in configuration even if that zone is registered with a different organization
6	Sirit Device behaves same in both AUTO and Semi AUTO modes.
7	Users in child enterprise organizations are not allowed to change performance tuning properties. The Performance Tuning properties are available to root organization only.
8	Admin user is able to create Alert Events and sensor events in the child organization. This is not the expected behavior.
9	Multiple Sensornets. Getting Action ScriptError while adding users
10	Multiple Sensornets. Changes in performance tuning properies in one sensornet are not reflecting in other sensornet
11	Multiple Sensornets. Changes in Event handler association in one sensornet are not reflecting in other sensornet
12	Edgeprocess Editor. Too much space left blank while creating the edges

2 Installation Manual

2.1 Introduction

This document describes the RFID IApp™ Installation program and step-by-step installation instructions.

2.2 Minimum Hardware and Software Requirements

A minimum of 8-bit color depth (256 colors) is required to run the RFID IApp installer. Other Hardware & Software requirement for RFID IApp™ components are listed below:

Sensor Net Manager

- Windows XP Professional for POC and Development environments / Windows Server 2008 64 bit for production use.
- Quad Core Processor with hyper threading.
- 8 GB RAM
- 3 x 140 GB SAS Hard Disk

Process Zone

- Windows XP Professional or Windows Server 2008
- Quad Core Processor (preferably with hyper threading)
- 4 GB RAM
- 2 x 40 GB SAS Hard Disk

Other General Requirements

- IE 6.0/7.0
- Macromedia Flash 9
- Screen resolution of 1280 x 1024 pixels.
- To view .XLS reports, MS Excel or Star Office must be installed in the installation system and the viewing system.
- To view .CSV files, MS Excel or another spreadsheet program capable of displaying comma-delimited files must be installed in the installation system and the viewing system.
- To use email alerts, a mail server must be available on the network.

2.3 iApp Software Support

Java:

- Java JDK Environment 1.5.0.11 for 32bit
- Java JDK Environment 1.5.0.22 for 64bit

Web Application Server:

- Tomcat 5.5.9 for 32bit
- Tomcat 5.5.28 for 64 bit

- Websphere 6.1
- Netweaver 7.1

Database:

- MS SQL Server Standard & above – 2005/2008
- Oracle Standard & above 10g
- IBM – DB2 Enterprise edition 9.7
- My SQL Community or Enterprise edition 5.0/5.1

Operating System:

- Windows XP Professional
- Windows 7 Professional 32 bit / 64 bit
- Windows 2003 Server
- Windows 2008 Server 32 bit / 64 bit

Flash:

- Flash 9.0.246 above

2.4 Pre-Requisites

RFID IApp requires the following software to be installed for a successful installation.

1. Java JDK Environment 1.5.0.11

Note: If JDK is also installed along with JRE make sure that JAVA_HOME environment variable is set to the JDK installation root directory and not to JRE. Also copy tools.jar from <java_install_dir>/lib to <jre_install_dir>/lib.

2. Database Software installed and tested for successful connectivity. The supported databases are listed in section 2

Note: For **Oracle** Installations it is required that:

- The Database and Users are created by the Database Administrator to be used during the installation and required privileges be granted to the Users.
- It is recommended that SYSTEM tablespace is not used and specific tablespaces be created and assigned to the database users.
- The Database is created with parameters CHARACTER SET and NATIONAL CHARACTER SET with values AL32UTF8 and AL16UTF16 respectively to accommodate multi language support

Note: For IBM –UDB installations it is required that

- The Database and Users are created by the Database Administrator to be used during the installation and required privileges be granted to the Users.

- The page size for the database created should be 32k

3. Jakarta Tomcat 5.5.9

2.5 Installing RFID iApp™

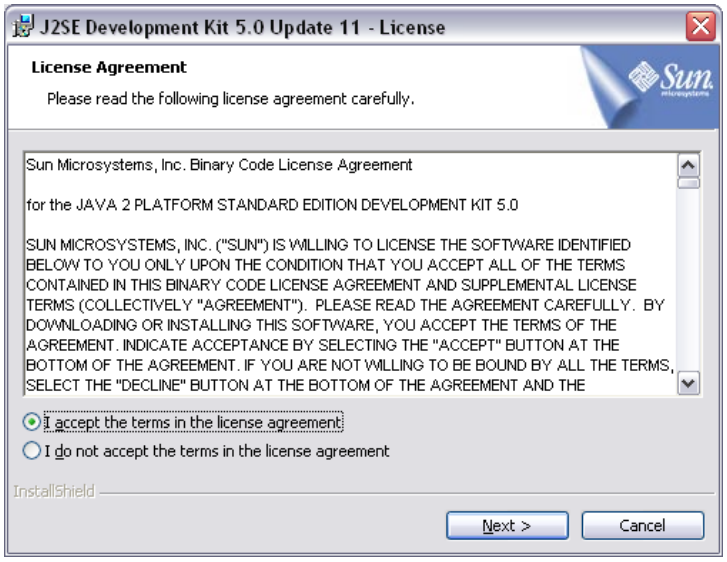
To install RFID IApp™, you need to install the applications in the order mentioned below:

1. Install the **Java environment 1_5_0_11**
2. Install the database and test for connectivity. Check the respective Database documentation for instructions in creating the database and users.
3. Install the **Jakarta-Tomcat 5.5.9 environment**.
4. Install **RFID IApp™**.

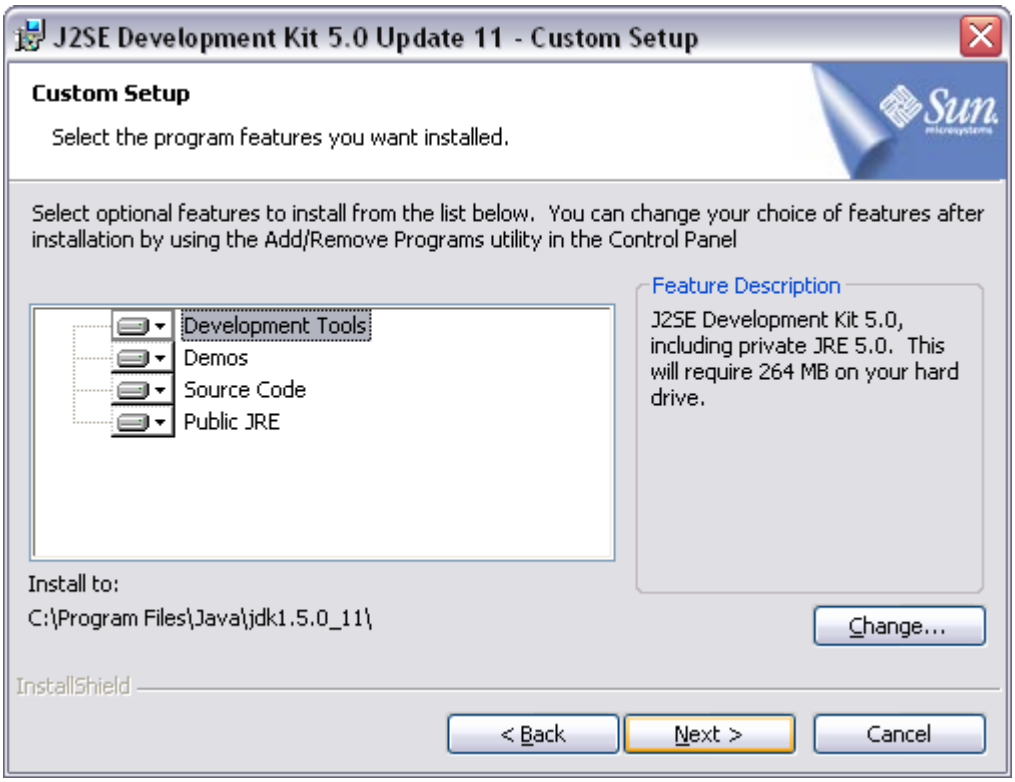
2.5.1 Installing the Java Environment

To install the Java environment **1_5_0_11:**

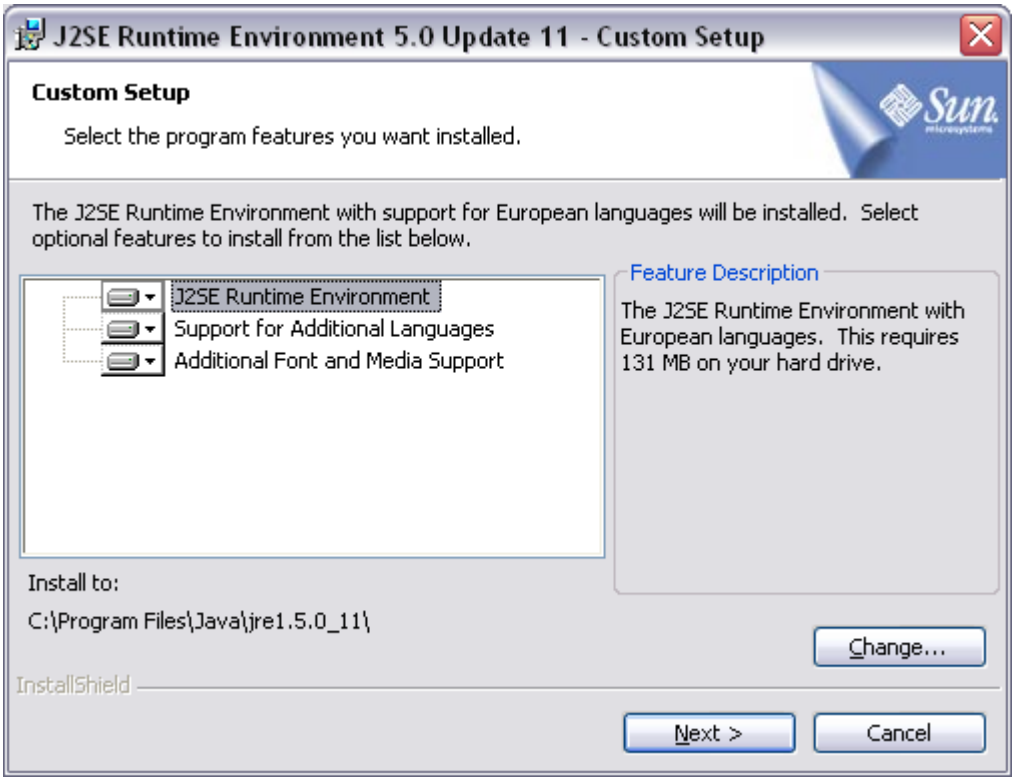
1. Download jdk-1_5_0_11-windows-i586-p.exe and double click the program.
2. Accept the license agreement and click Next.


J2SE Development Kit 5.0 Update 11 - License
License Agreement
Please read the following license agreement carefully.
Sun Microsystems, Inc. Binary Code License Agreement
for the JAVA 2 PLATFORM STANDARD EDITION DEVELOPMENT KIT 5.0
SUN MICROSYSTEMS, INC. ("SUN") IS WILLING TO LICENSE THE SOFTWARE IDENTIFIED BELOW TO YOU ONLY UPON THE CONDITION THAT YOU ACCEPT ALL OF THE TERMS CONTAINED IN THIS BINARY CODE LICENSE AGREEMENT AND SUPPLEMENTAL LICENSE TERMS (COLLECTIVELY "AGREEMENT"). PLEASE READ THE AGREEMENT CAREFULLY. BY DOWNLOADING OR INSTALLING THIS SOFTWARE, YOU ACCEPT THE TERMS OF THE AGREEMENT. INDICATE ACCEPTANCE BY SELECTING THE "ACCEPT" BUTTON AT THE BOTTOM OF THE AGREEMENT. IF YOU ARE NOT WILLING TO BE BOUND BY ALL THE TERMS, SELECT THE "DECLINE" BUTTON AT THE BOTTOM OF THE AGREEMENT AND THE
I accept the terms in the license agreement
I do not accept the terms in the license agreement
InstallShield
Next >
Cancel

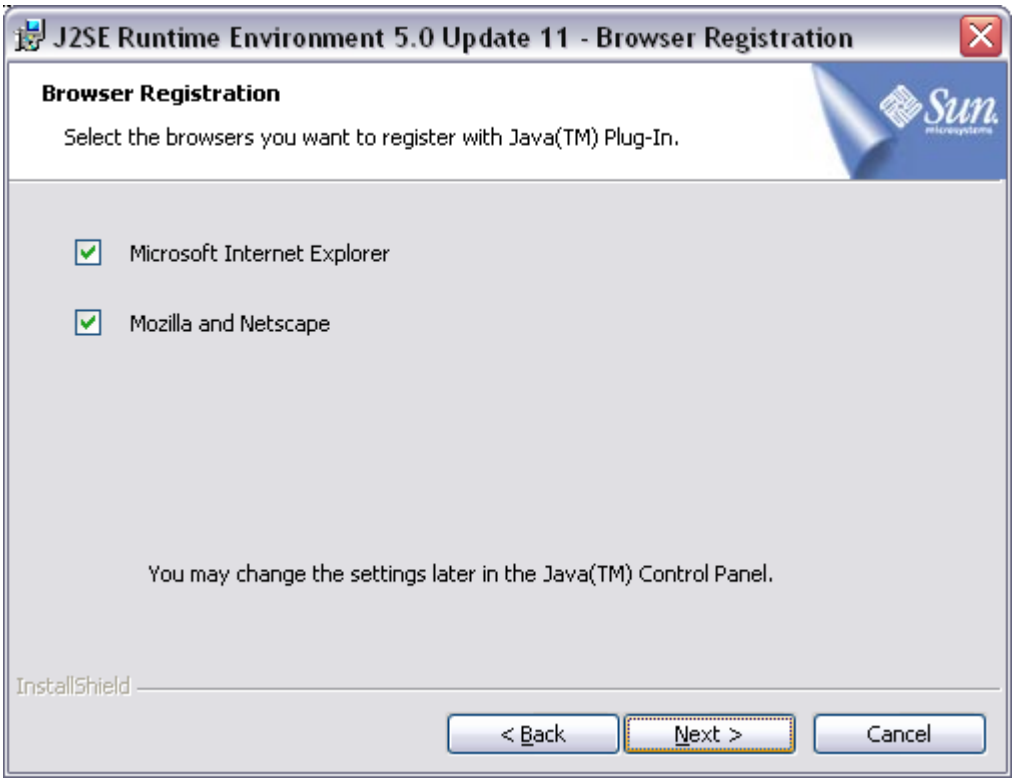
1. Change the install path to install at a different path than the default path. Click Next.


J2SE Development Kit 5.0 Update 11 - Custom Setup
Custom Setup
Select the program features you want installed.
Select optional features to install from the list below. You can change your choice of features after installation by using the Add/Remove Programs utility in the Control Panel
Development Tools
Demos
Source Code
Public JRE
Feature Description
J2SE Development Kit 5.0, including private JRE 5.0. This will require 264 MB on your hard drive.
Install to:
C:\Program Files\Java\jdk1.5.0_11\
Change...
InstallShield
< Back
Next >
Cancel

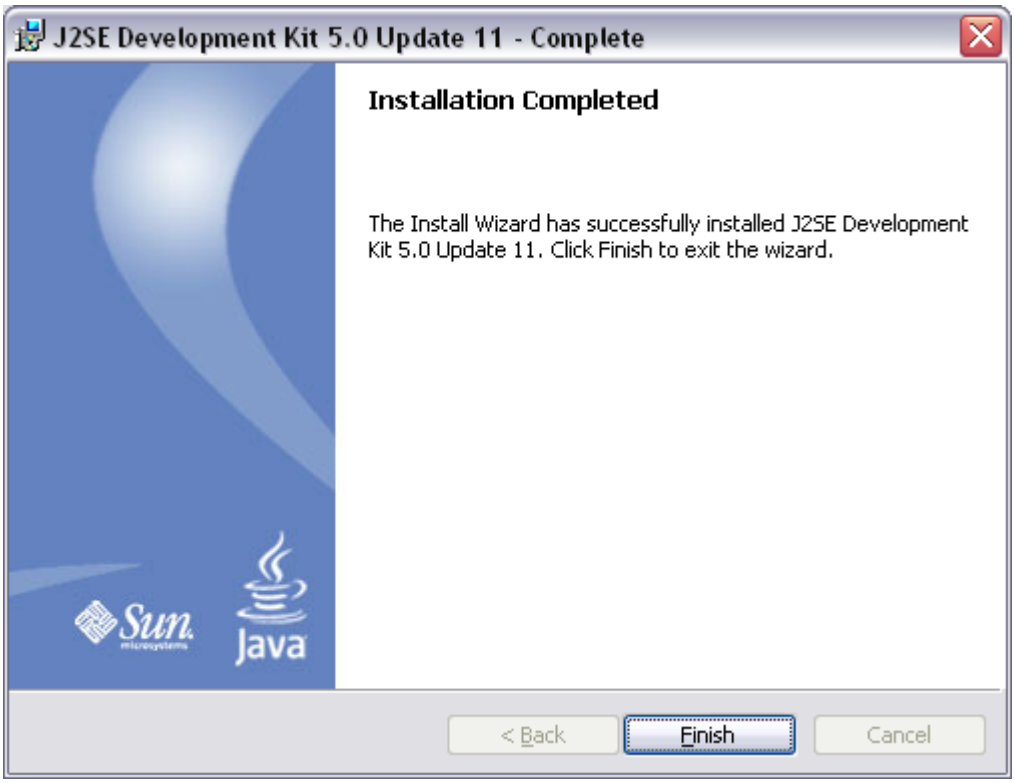
2. For the J2SE run time environment click Browse to change the install path or leave the default path. Click Next.


J2SE Runtime Environment 5.0 Update 11 - Custom Setup
Custom Setup
Select the program features you want installed.
The J2SE Runtime Environment with support for European languages will be installed. Select optional features to install from the list below.
J2SE Runtime Environment
Support for Additional Languages
Additional Font and Media Support
Feature Description
The J2SE Runtime Environment with European languages. This requires 131 MB on your hard drive.
Install to:
C:\Program Files\Java\jre1.5.0_11\
Change...
InstallShield
Next >
Cancel

3. Select the browser to register and click Next


J2SE Runtime Environment 5.0 Update 11 - Browser Registration
Browser Registration
Select the browsers you want to register with Java(TM) Plug-In.
Sun
Microsoft Internet Explorer
Mozilla and Netscape
You may change the settings later in the Java(TM) Control Panel.
InstallShield
< Back
Next >
Cancel

4. Click Finish to complete the installation.


J2SE Development Kit 5.0 Update 11 - Complete
Installation Completed
The Install Wizard has successfully installed J2SE Development Kit 5.0 Update 11. Click Finish to exit the wizard.
Sun
Java
< Back
Finish
Cancel

The Java environment is now installed on your machine.

Note: If JDK is also installed along with JRE make sure that JAVA_HOME environment variable is set to the JDK installation root directory and not to JRE. Also copy tools.jar from <java_install_dir>/lib to <jre_install_dir>/lib.

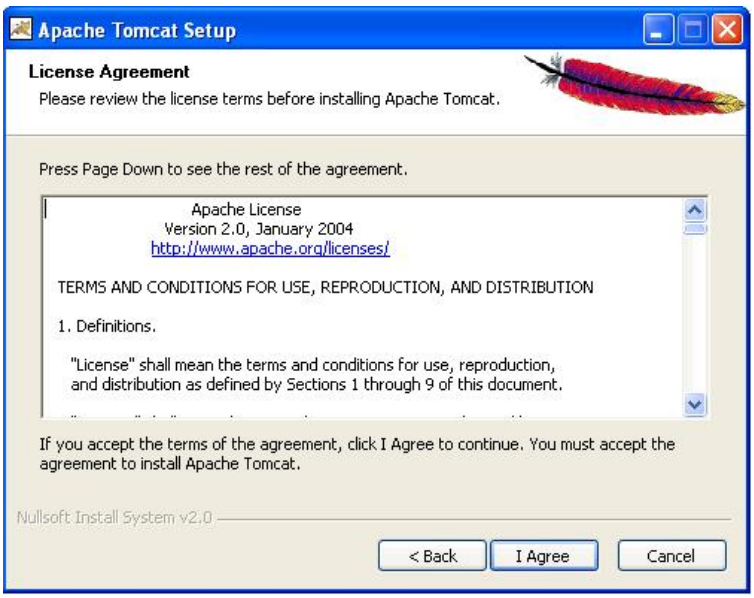
2.5.2 Installing Jakarta-Tomcat 5.5

To install Tomcat 5.5:

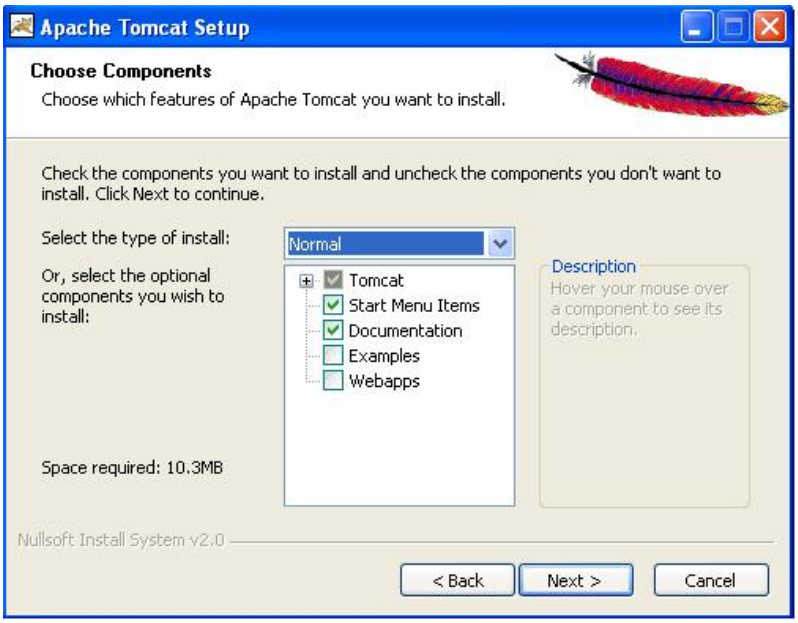
1. Download Jakarta-tomcat-5.5.9.exe and Double-click.
2. Click Next.

Apache Tomcat Setup
The Apache Jakarta Project
http://jakarta.apache.org/
Welcome to the Apache Tomcat Setup Wizard
This wizard will guide you through the installation of Apache Tomcat.
It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.
Click Next to continue.
Next >
Cancel

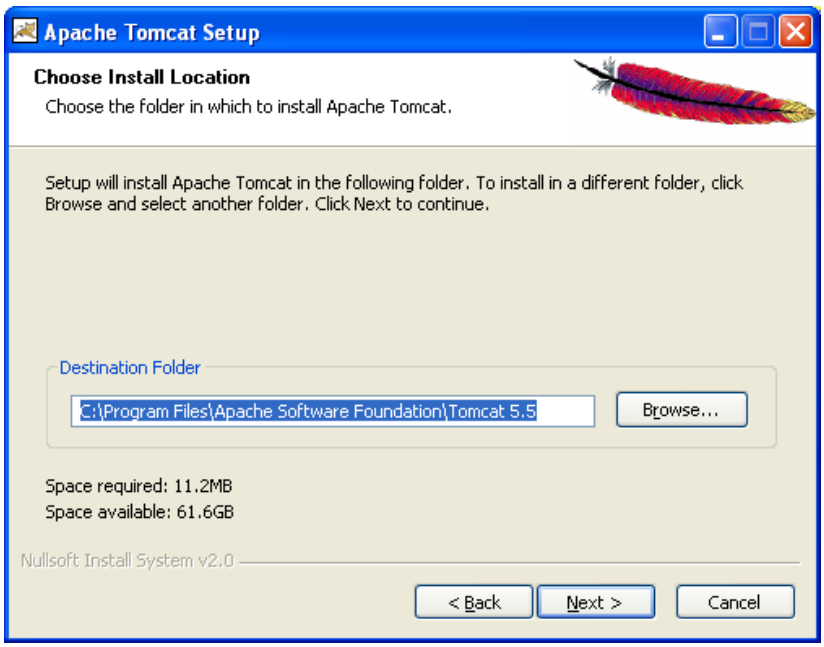
3. Click I Agree.


Apache Tomcat Setup
License Agreement
Please review the license terms before installing Apache Tomcat.
Press Page Down to see the rest of the agreement.
Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/
TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION
1. Definitions.
"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.
If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Apache Tomcat.
Nullsoft Install System v2.0
< Back
I Agree
Cancel

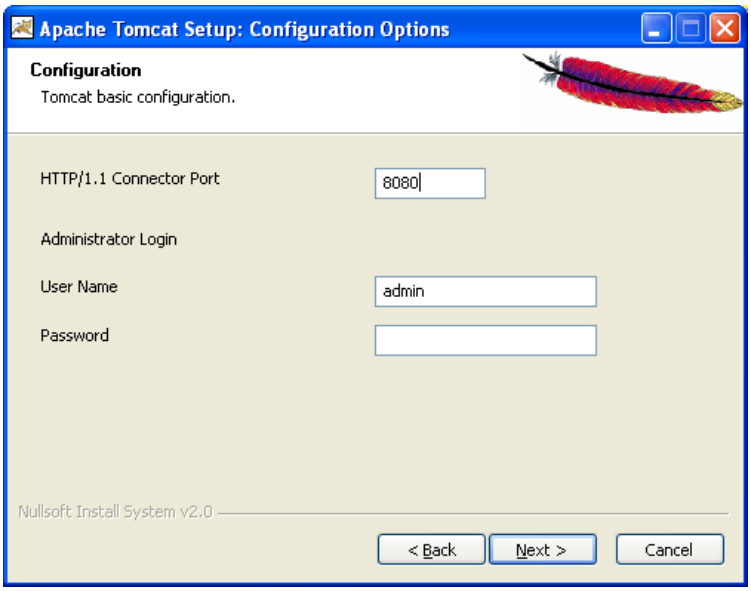
4. Click Next.


Apache Tomcat Setup
Choose Components
Choose which features of Apache Tomcat you want to install.
Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.
Select the type of install:
Normal
Or, select the optional components you wish to install:
Tomcat
Start Menu Items
Documentation
Examples
Webapps
Description
Hover your mouse over a component to see its description.
Space required: 10.3MB
Nullsoft Install System v2.0
< Back
Next >
Cancel

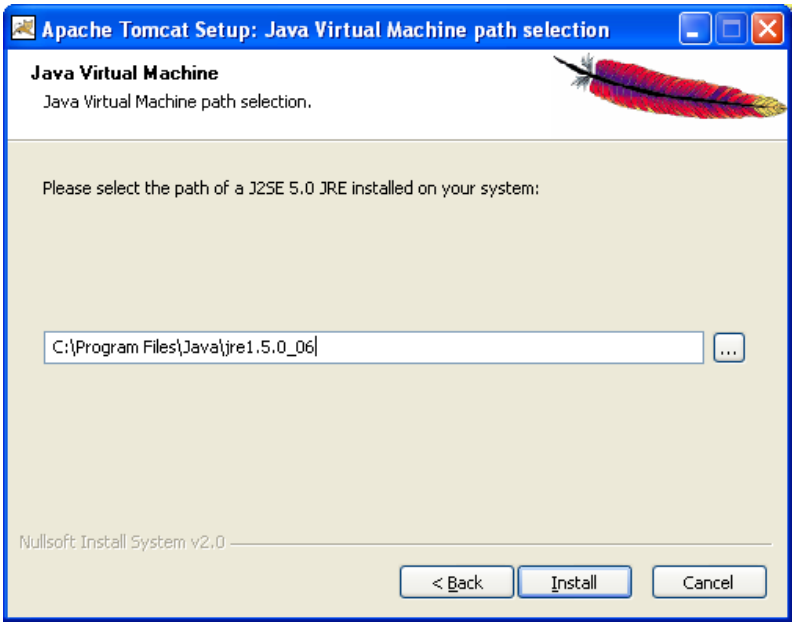
5. You can install the Tomcat in the default location or select the destination folder by clicking Browse. Click Next.


Apache Tomcat Setup
Choose Install Location
Choose the folder in which to install Apache Tomcat.
Setup will install Apache Tomcat in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.
Destination Folder
C:\Program Files\Apache Software Foundation\Tomcat 5.5
Browse...
Space required: 11.2MB
Space available: 61.6GB
Nullsoft Install System v2.0
< Back
Next >
Cancel

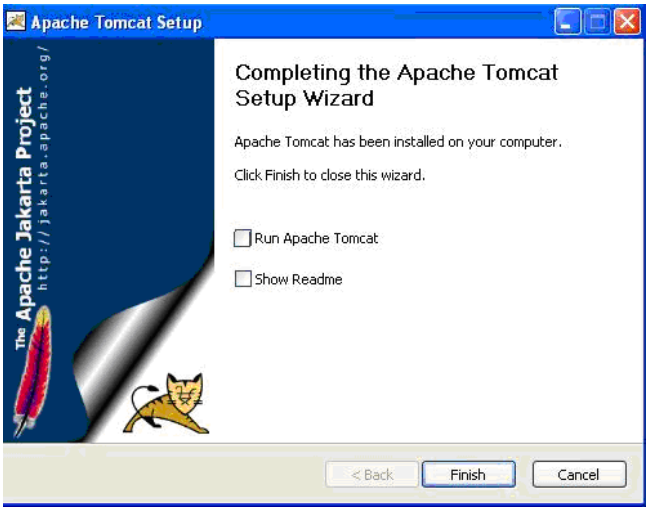
6. Specify 8080 in the HTTP/1.1 Connector Port field. Leave the User Name and Password fields as it is. Click Next. Make sure that the Port number specified is not used by any other service.



7. Leave the Java Virtual Machine path as it is and click Install.


Apache Tomcat Setup: Java Virtual Machine path selection
Java Virtual Machine
Java Virtual Machine path selection.
Please select the path of a J2SE 5.0 JRE installed on your system:
C:\Program Files\Java\jre1.5.0_06
...
Nullsoft Install System v2.0
< Back
Install
Cancel

8. Once setup is complete, you will see the following screen. You can select the option that is relevant for you to either run Tomcat or display the readme.



9. Click Finish.

10. Open <Tomcat_Install_Dir>/Bin/tomcat5w.exe. Add the following entry in the Java Tab – Java Options of the properties window.

    -Dcom.sun.management.jmxremote=true

    -Dcom.sun.management.jmxremote="catalina"

    -Dcom.sun.management.jmxremote.port=8081

    -Dcom.sun.management.jmxremote.ssl=false

    -Dcom.sun.management.jmxremote.authenticate=false

    XX:+HeapDumpOnOutOfMemoryError

    -XX:MaxPermSize=512M

    -Xms512m

    -Xmx512m

This completes Tomcat Installation.

***Note**: It is recommended that the java heap size is set to 50% of the physical memory available on the server. The java virtual machine is set to the server jvm.dll.*

2.5.3 Installing RFID iApp™ Application

The RFID IApp™ application has the following components. You should install RFID IApp™ components in the order mentioned below.

- **Sensor Net Manager**: GUI to manage one or more Process Zones.
- **Process Zones**: Backend processor to read data from readers and process it further.
- **Print Server**

RFID IApp™ ships with a graphical installation process to aid in installing various components and examples. Next few sections describe the installation process for different components. Depending on the component you want to install, read the appropriate section.

There are different deployment options based on the requirements and the configuration of the Servers.

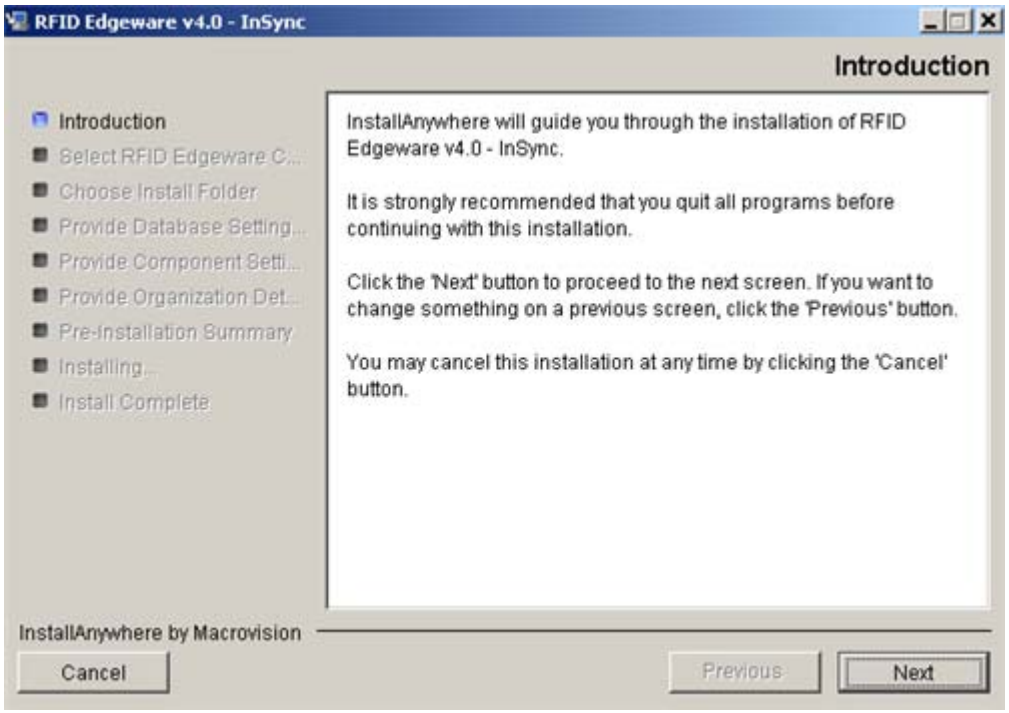
- All components can be installed on a single machine.
- Each component can be installed on different machines.
- The databases used by the components can be on the same machine as the component or on separate machines.
- One could share a machine for certain components.

Note: **If there is a previous version of IApp running on the server where the new installation is being done make sure that the localhost folder is deleted from the tomcat installation directory located at <tomcat_install_dir>/work/Catalina before the newly deployed product is run.**

2.5.4 Installing Sensor Net Manager

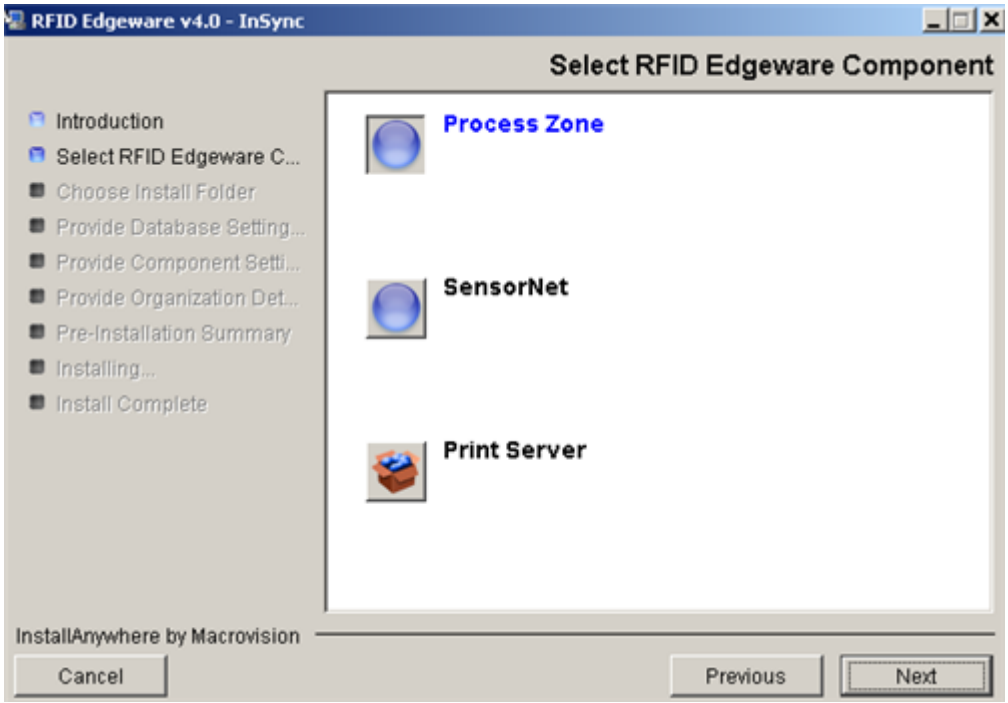
To install Sensor Net Manager:

1. Double click the install.exe file. This will launch the installation program as shown below:

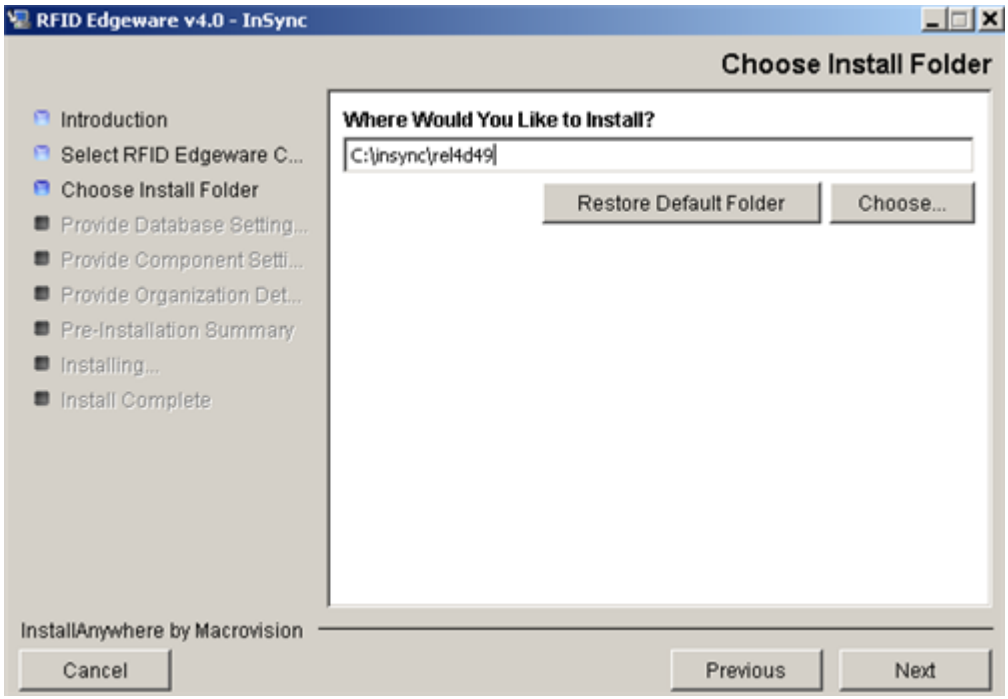

RFID Edgeware v4.0 - InSync
Introduction
Introduction
Select RFID Edgeware C...
Choose Install Folder
Provide Database Setting...
Provide Component Setti...
Provide Organization Det...
Pre-Installation Summary
Installing...
Install Complete
InstallAnywhere will guide you through the installation of RFID Edgeware v4.0 - InSync.
It is strongly recommended that you quit all programs before continuing with this installation.
Click the 'Next' button to proceed to the next screen. If you want to change something on a previous screen, click the 'Previous' button.
You may cancel this installation at any time by clicking the 'Cancel' button.
InstallAnywhere by Macrovision
Cancel
Previous
Next

2. Click Next.

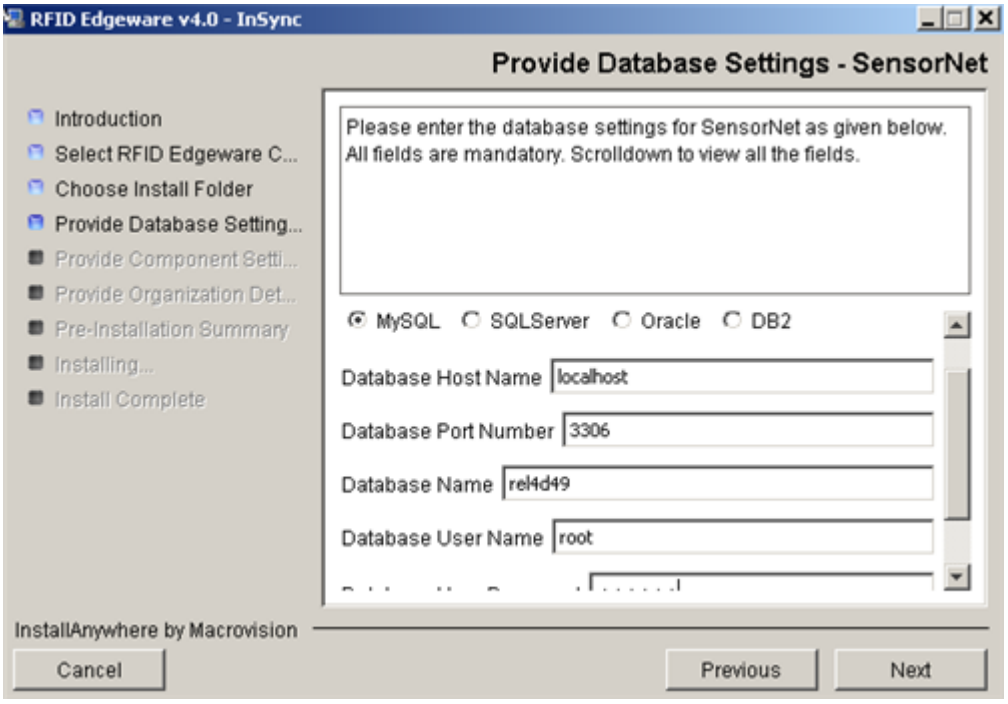
3. Select Sensornet and click Next.



4. You can install the component in the default location or select the destination folder by clicking **Choose**.



5. Click **Next**.
6. Select the database. You may select MySql or SQL Server or Oracle as your database. Specify the Database settings as below:



Note: The installation software will connect to the database using the information provided. Please validate the data using the Database client provided by the vendor to connect and access it, The installation will fail if there is an issue with connection i.e. not having the driver or physical connection or the credentials or wrong. The installation directory needs to be deleted before installing again if the installation fails.

7. Select the relevant database. Specify the Database settings. It is recommended that the Database connectivity is tested before proceeding with this step.

For MySql, MS-Sql Server and Oracle installations the following are required

- **Database Host Name**: Machine/Server Name where the database is running. If it is running on the same machine as RFID IApp, then leave this field as “localhost”.
- **Database Port Number**: Enter the port number of the machine where database is running.
- **Database Name:** Name of the database used for Sensornet.
- **Database User Name**: User Name to access the above database.
- **Database User Password**: Password for the above user.
- **Current Schema (DB2):** Leave it blank

Note for MySql and Sql Server*: In case the database does not exist, the installer creates the database with the name provided if you have the permission to create databases. Typically, the “root” user has the privileges to create databases. Please check with your database administrator if you are unable to create a database.*

Note for Oracle*: It is required that the database and users are pre-created before you start the installation. Please check the Oracle documentation for information on creating the database and Users. The Users created should have the privileges to create tables. The Installer does not create the database and the users if they do not exist.*

It is recommended that SYSTEM tablespace is not used and specific tablespaces be created and assigned to the database users.

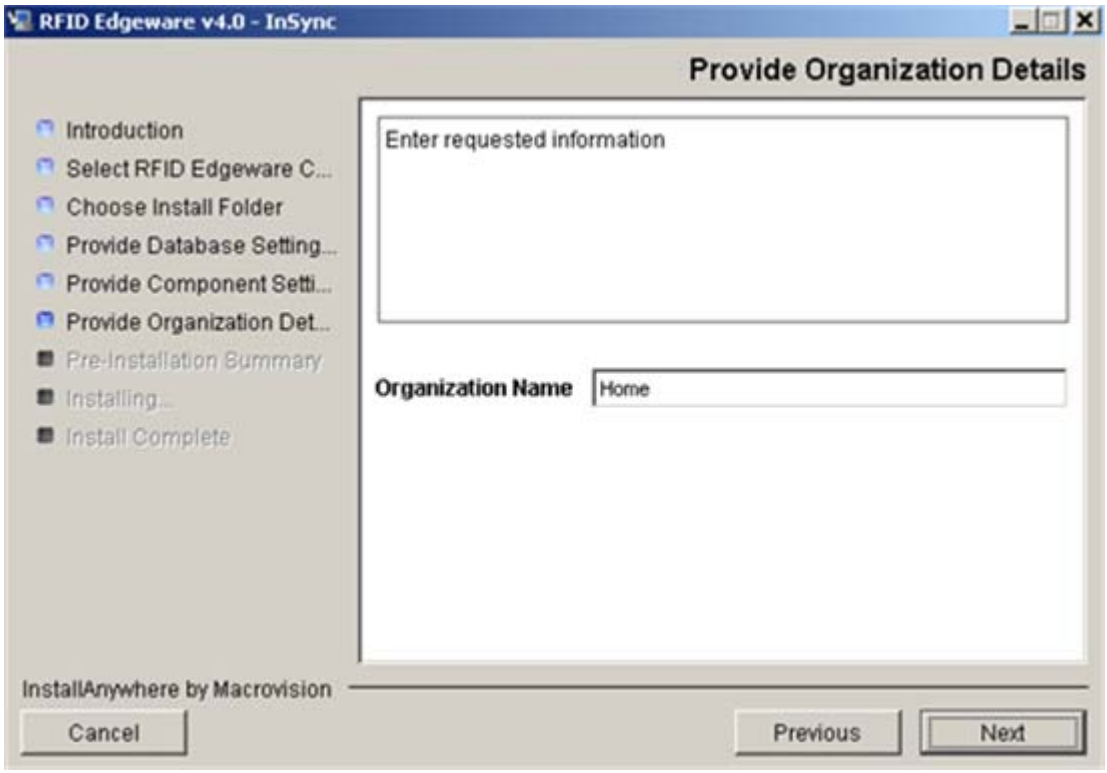
Also, as mentioned earlier, while creating the Oracle database, we need to provide the parameters CHARACTER SET with a value of AL32UTF8, and provide NATIONAL CHARACTER SET with a value of AL16UTF16. This is done to accommodate multi language support in the application.

For DB2 installations the following is required

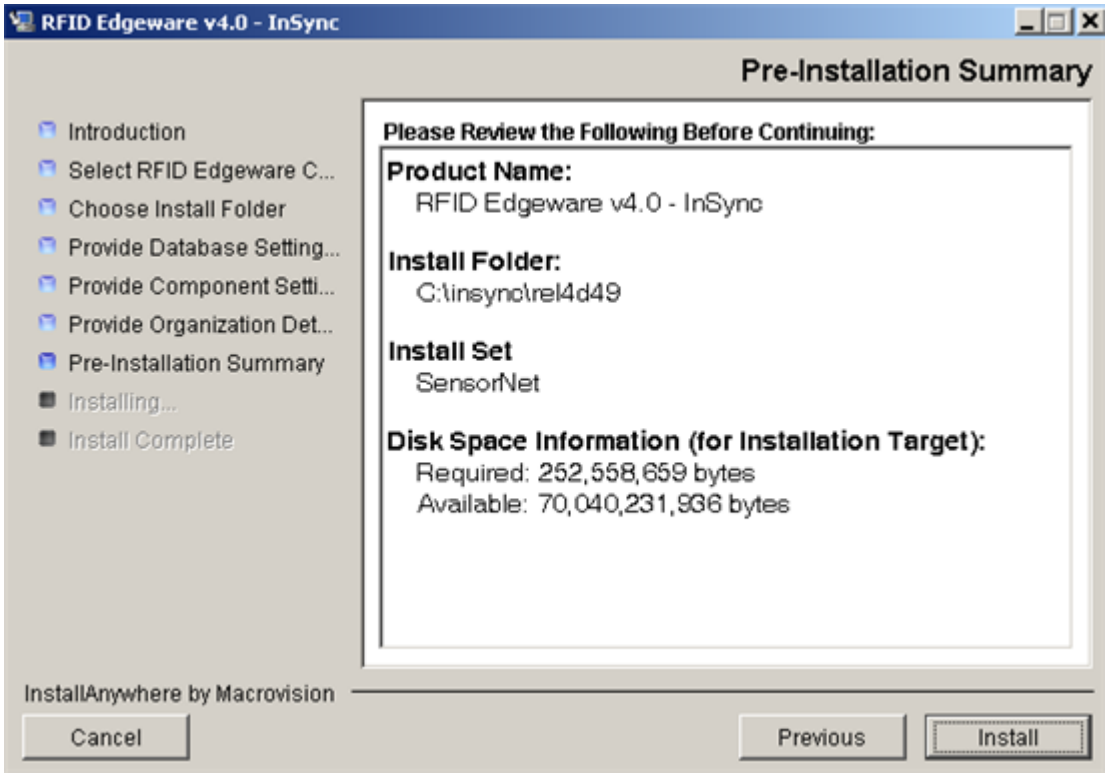
- **Database Host Name**: Machine/Server Name where the database is running. If it is running on the same machine as RFID IApp, then leave this field as “localhost”.
- **Database Port Number**: Enter the port number of the machine where database is running.
- **Database Name:** Name of the database used for Sensornet.
- **Database User Name**: User Name to access the above database.
- **Database User Password**: Password for the above user.
- **Current Schema (DB2):** is the authorization ID of the current DB2 session user. The DB2 administrator should be able to give this information.

Note for DB2*: It is required that the database and users are pre-created before you start the installation. Please check the DB2 documentation for information on creating the database and Users. The Users created should have the privileges to create tables. The Installer does not create the database and the users if they do not exist.*

8. Click **Next**.
9. Enter the Home Enterprise Organization name. It is set to ‘Home’ by default. This is the name of the organization that is going to manage/use the application.

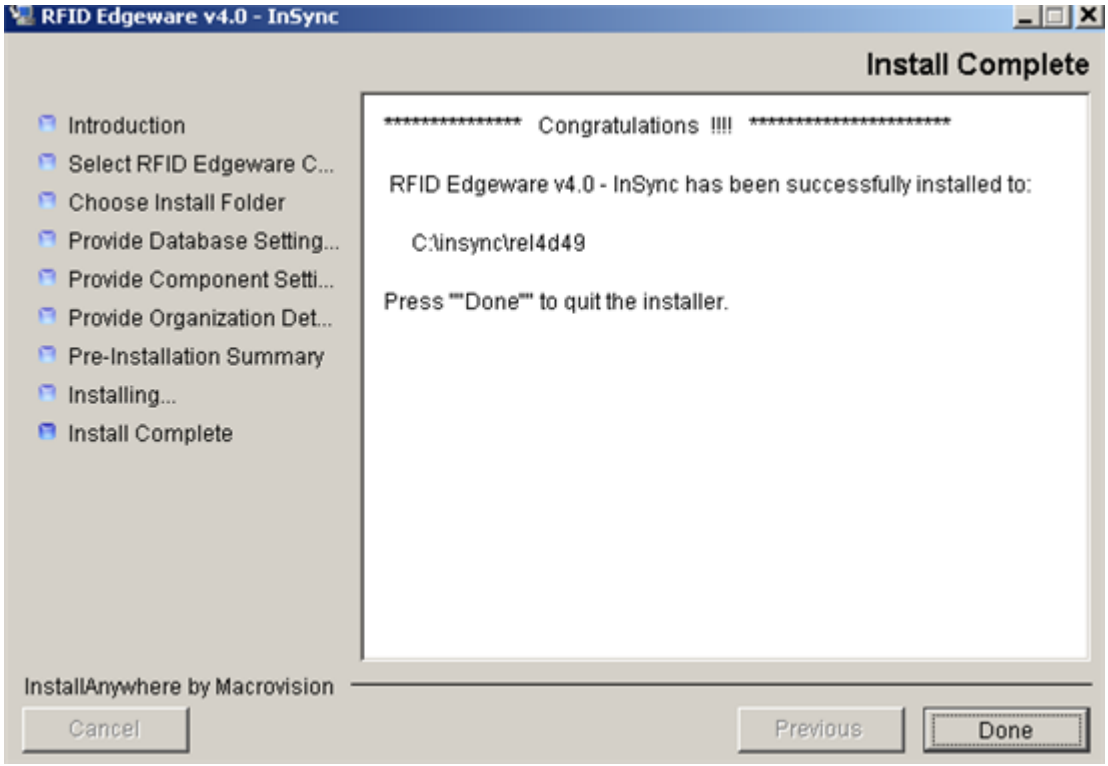


10. Click Next
11. Now that all configuration details are entered, a confirmation is displayed with all the settings. Use this confirmation to verify the settings.



12. Click Install. Installation will take a few minutes. Wait for the confirmation window.The Install scipts takes some time to load some default data. The following screen will be visible

13. When installation is complete, click **Done** to finish the installation. After successful installation the following screen is shown. In case of failure delete the installation folder and check the software versions supported by IApp. The document is available along with this installation document. You will need JDK and not JRE when using the installer. The installation log with name “RFID_Edgeware_v4.0_-_InSync_InstallLog.log” is saved in the installation directory . The log can provide additional information incase of failed installation.



The RFID IApp™ installation is complete for Sensor Net Manager.

Sensornet Post Installation Steps

1. Edit server.xml file in tomcat server. The file is present in the <Tomcat_install_dir>\Tomcat 5.5\conf, Add the following entry:

```
<Context path="/sensornet"
                docBase="<install_directory>/RFIDIApp/sensornet/app"
                crossContext="false"
                debug="0"
                reloadable="false" >
                <Manager className="org.apache.catalina.session.PersistentManager"
saveOnRestart="false"/>
                </Context>
```

Where docBase is the path of the app folder in sensornet installation.

Note: The context path would be the same as provided in the instance registry during configuration.For more information check the Administration manual.

2. Copy sensornet.license.lic file to "<install_directory>\RFIDIApp\sensornet\app\WEB-INF\classes" folder. This provided by InSync Software Inc. In order to generate the license install provides a tool to collect the systems signature and generate the license file.

3. Set the tomcat parametes. Locate the Tomcat5w.exe file in the directory "C:\Program Files\Apache Software Foundation\Tomcat 5.5\bin" and double click on it. Set the following parameters

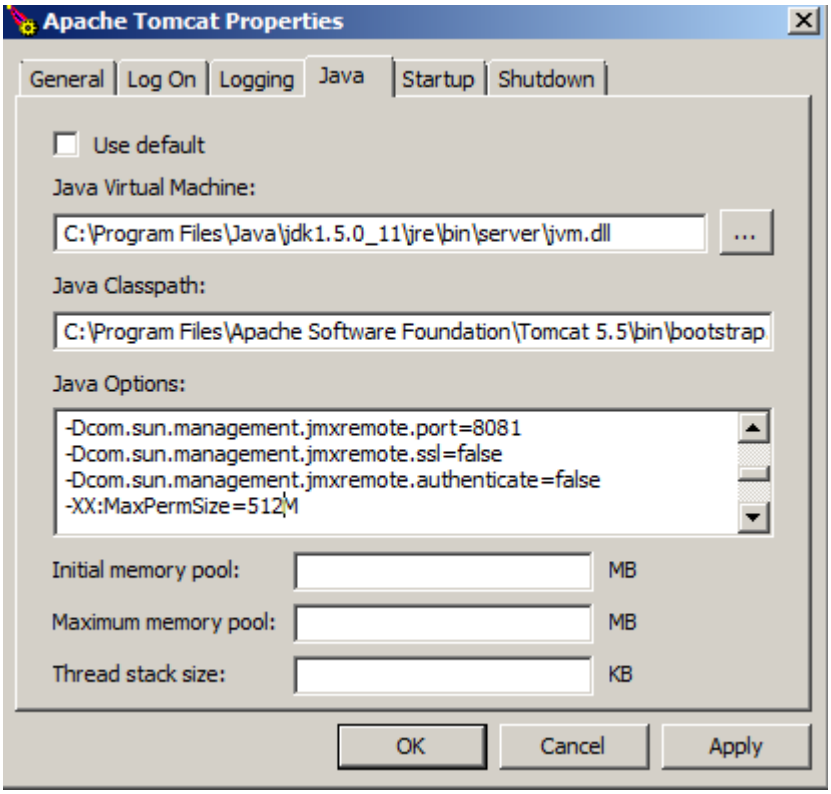
    i. Java properties: the following properties should be set

        Enable JConsole
        -Dcom.sun.management.jmxremote=true

-Dcom.sun.management.jmxremote="catalina"
-Dcom.sun.management.jmxremote.port=8081
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.authenticate=false

ii. Set memory settings
-XX:MaxPermSize=512M
-Xms1024m
-Xmx1024m
XX:+HeapDumpOnOutOfMemoryError

iii. Check the JVM path is set correctly

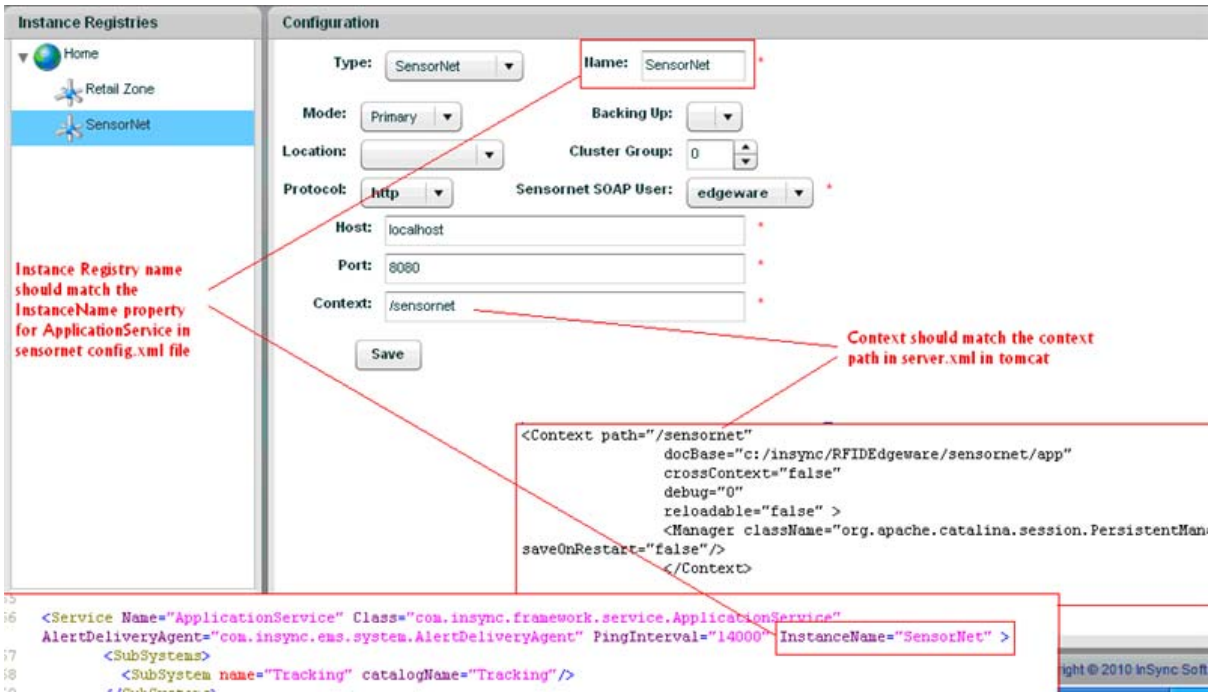


4. Start tomcat server where the SensorNet is configured

5. Login to sensornet using the URL http://<machine>:<portnumber>/<context_path>. At installation the system creates an Administrator User: Name: admin Password: admin

6. Navigate to Administration Menu > Instance Registry screen.

7. Ensure that the Instance Registry name is same as the value of the ‘InstanceName’ property of ApplicationService in the sensornet config.xml file. The file is available in ‘sensornet\app\WEB-INF’ folder.



8. Ensure that the value of ‘Context’ in SensorNet type of instance registry is same as the context path in the server.xml of tomcat (See step # 1

9. Ensure that Host and Port are correct in the SensorNet type of instance registry. This information will be used by other SensorNets to communicate with each other. The location is used to display in the central monitoring screen.

10. The mode should be set to “Primary” as hot standby is not supported in this release. The cluster is used to keep the cache synchronized. Another cluster can be added for central monitoring purpose.

11. Navigate to Configuration > Explorer screen

12. Create locations and processzone

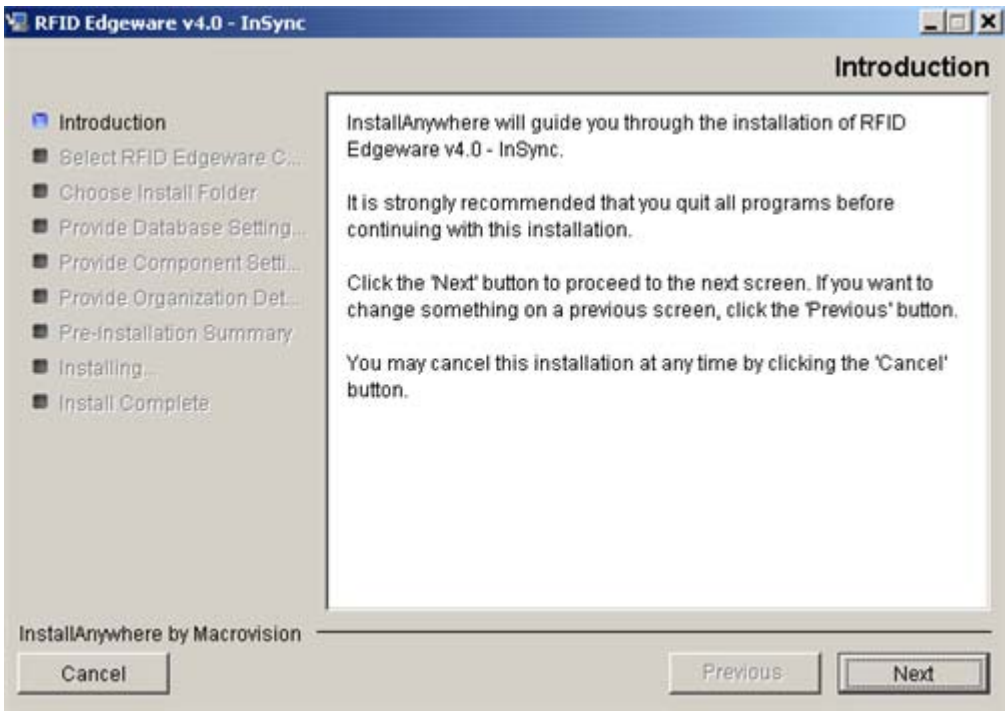
Note:

1. For more information on Instance Registry check the Administration Manual ->Instance Regsitry section.
2. For more information on Configuration check the Reference manual->Overview.

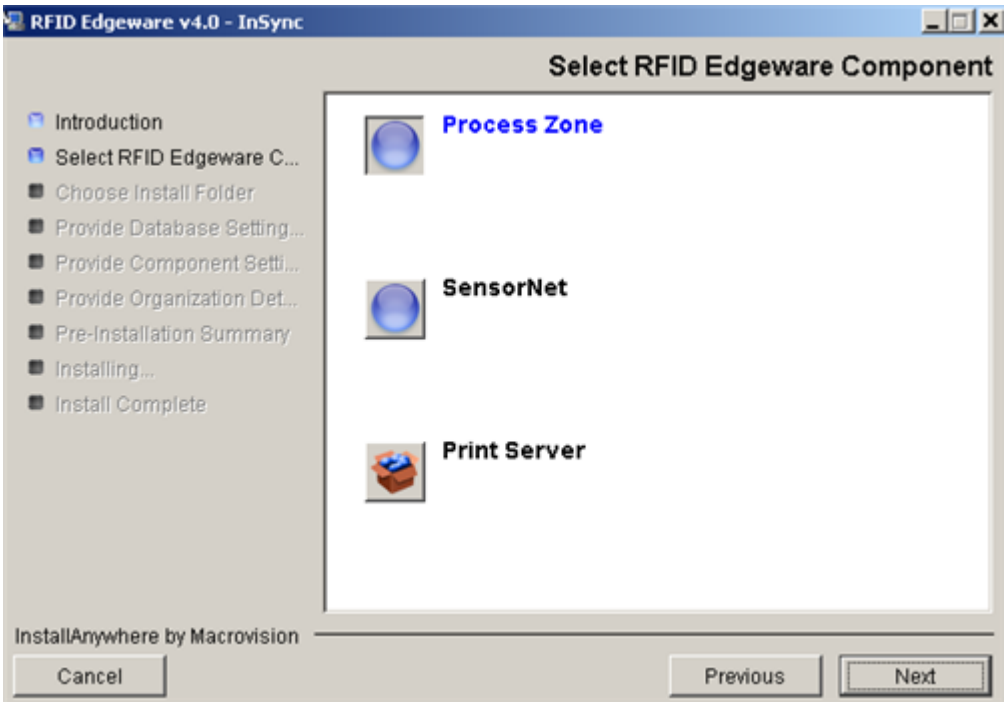
2.5.5 Installing Process Zone

To install the Process Zone:

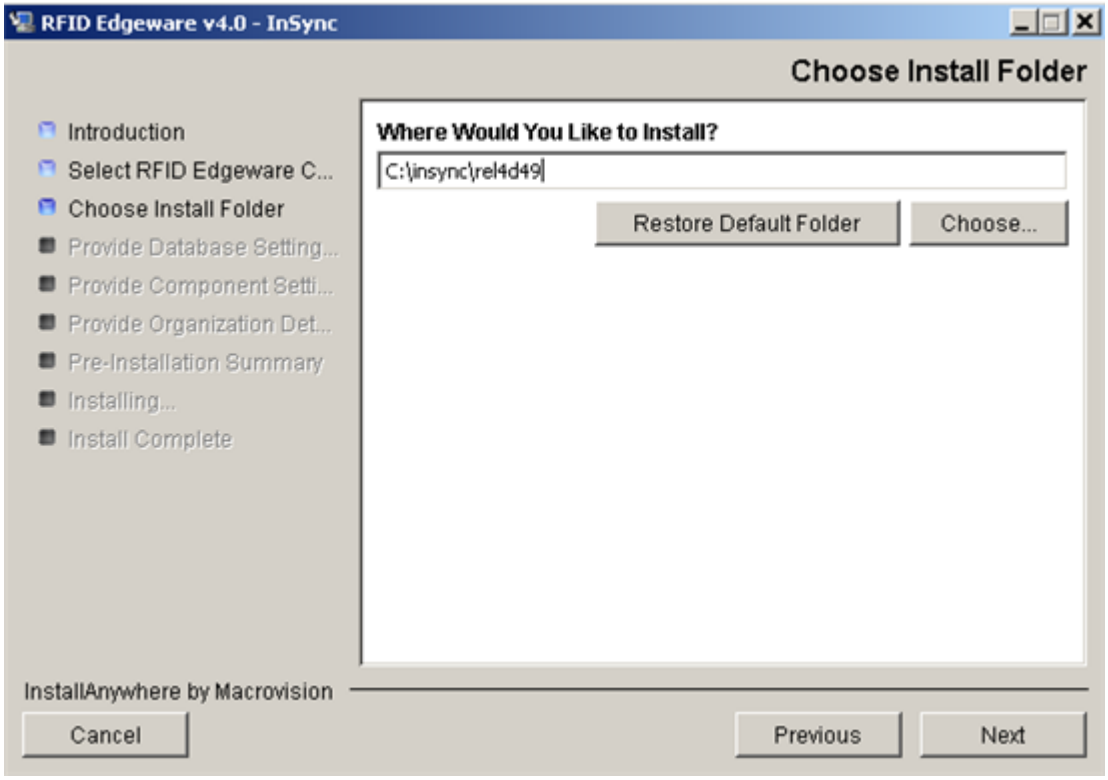
1. Double click the install.exe file . This will launch the installation program as shown below.


RFID Edgeware v4.0 - InSync
Introduction
Introduction
Select RFID Edgeware C...
Choose Install Folder
Provide Database Setting...
Provide Component Setti...
Provide Organization Det...
Pre-Installation Summary
Installing...
Install Complete
InstallAnywhere will guide you through the installation of RFID Edgeware v4.0 - InSync.
It is strongly recommended that you quit all programs before continuing with this installation.
Click the 'Next' button to proceed to the next screen. If you want to change something on a previous screen, click the 'Previous' button.
You may cancel this installation at any time by clicking the 'Cancel' button.
InstallAnywhere by Macrovision
Cancel
Previous
Next

2. Click Next.
3. Select Process Zone and click Next.



4. You can install the component in the default location or select the destination folder by clicking Choose.

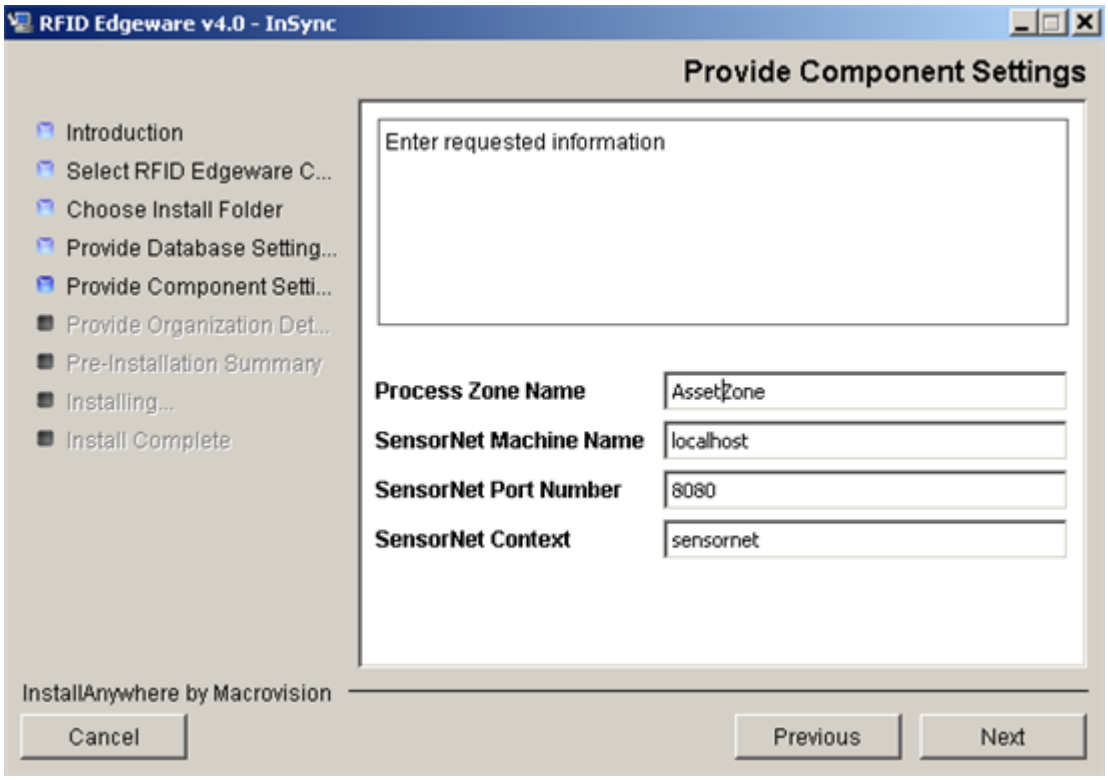


5. Click **Next**.
6. Specify the component settings as given below.
7. **Process Zone Name**: Name of the Process Zone to be installed. The process zone name cannot be ‘process zone’ since it is a reserved word.
8. **SensorNet Manager Machine Name**: Machine name or IP address where Sensornet is running.

Note: The ipaddress or machine name should be used instead of localhost

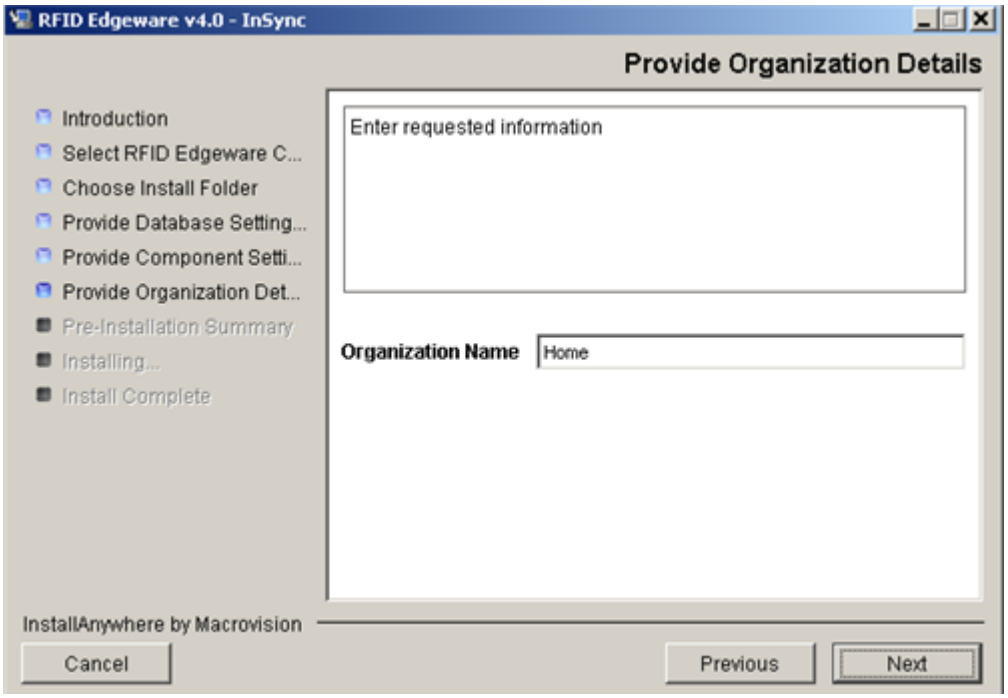
A. Sensornet Port Number: port number of the tomcat instance where Sensornet is running.

B. SensorNet Context: Context by which the Sensornet instance is identified. This is used while setting up the instance registry for sensornet and also in the Server.xml of tomcat instance running Sensornet.



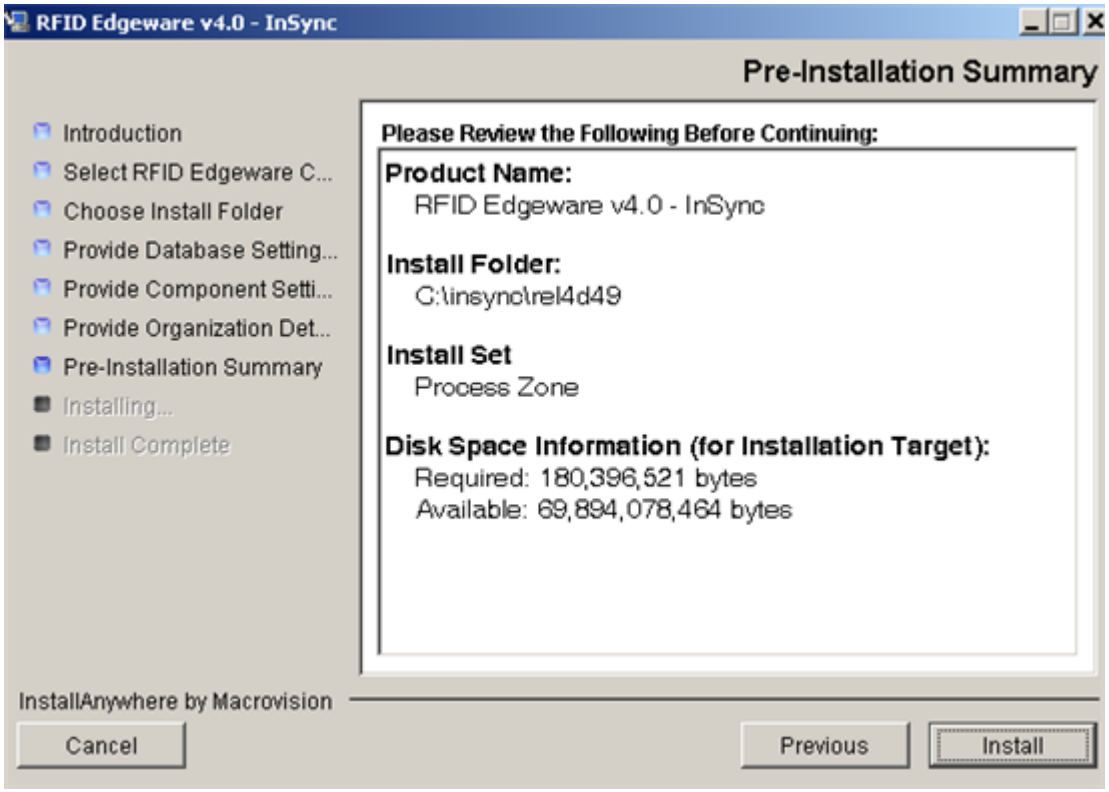
9. Click Next.

10. Enter the Enterprise Organization for which the Zone is getting created and installed. This is the name of the enterprise to which this zone belongs to and which is going to use the application.



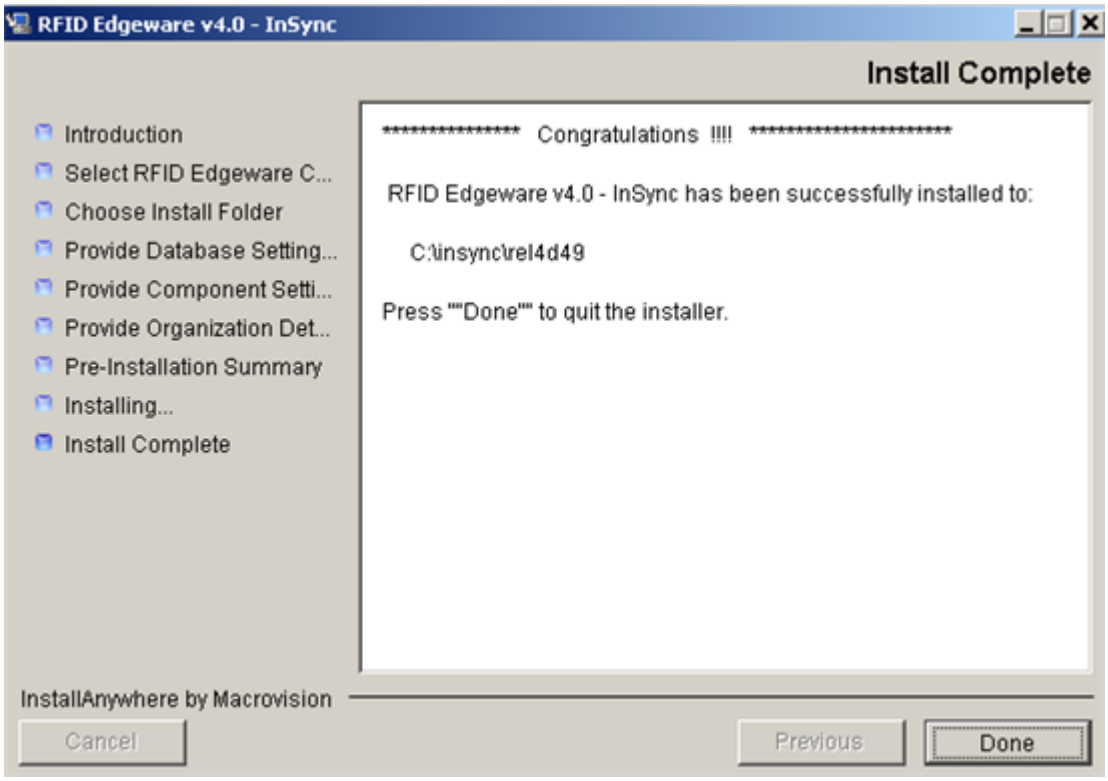
11. Click Next.

12. Now that all configuration details are entered, a confirmation is displayed with all the settings. Use this confirmation to verify the settings.



13. Click Install. Wait for the installation to complete.

14. Click Done to finish the installation. After successful installation the following screen is shown. In case of failure delete the installation folder and check the software versions supported by IApp. The document is available along with this installation document. You will need JDK and not JRE when using the installer.



The RFID IApp™ installation is complete for Process Zone.

Post Installation steps for Process Zone

1. Copy the processzone.license.lic file to '<processzone installation folder>\app\WEB-INF\classes' folder

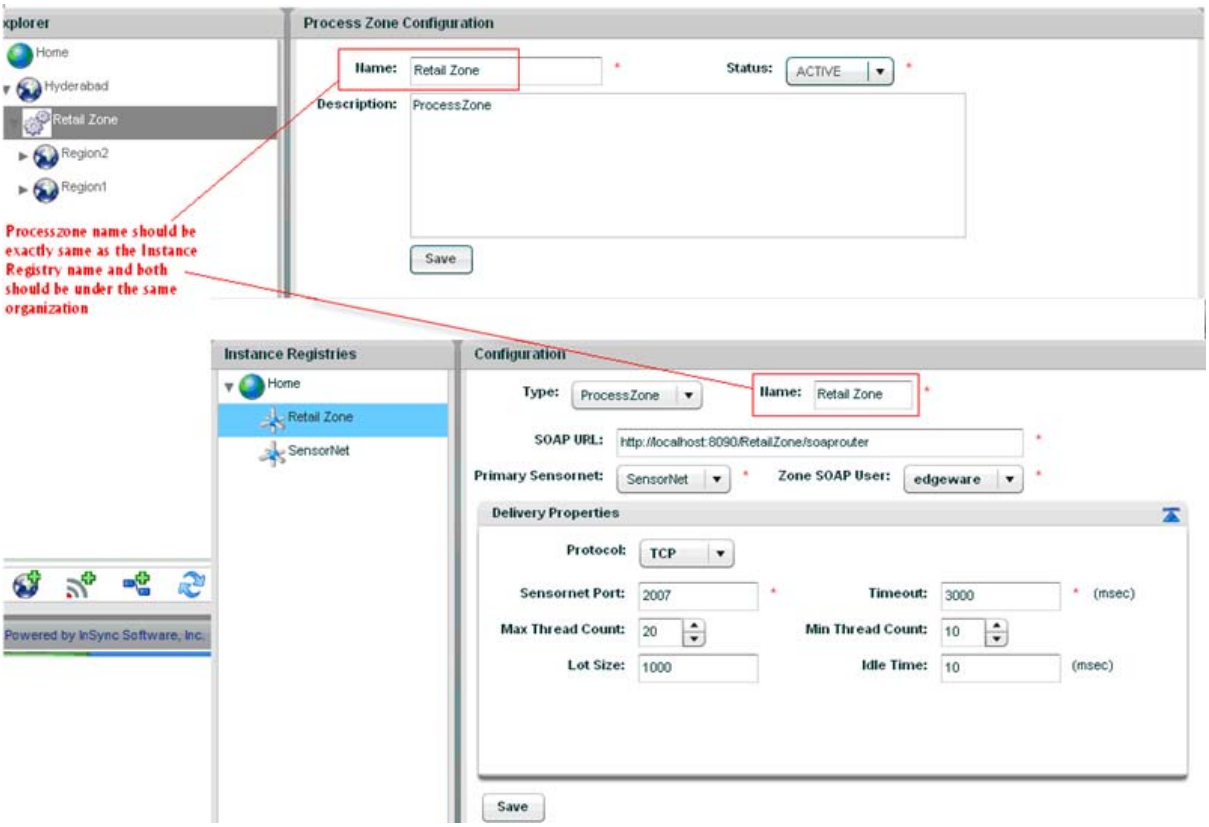
2. Edit server.xml file in tomcat server. Add the following entry:

```
<Context path="/RetailZone"
                        docBase="C:/insync/RFIDIApp/RetailZone/app"
                        crossContext="false"
                        debug="0"
                        reloadable="true" >
                        <Manager className="org.apache.catalina.session.PersistentManager"
saveOnRestart="false"/>
                        </Context>
```

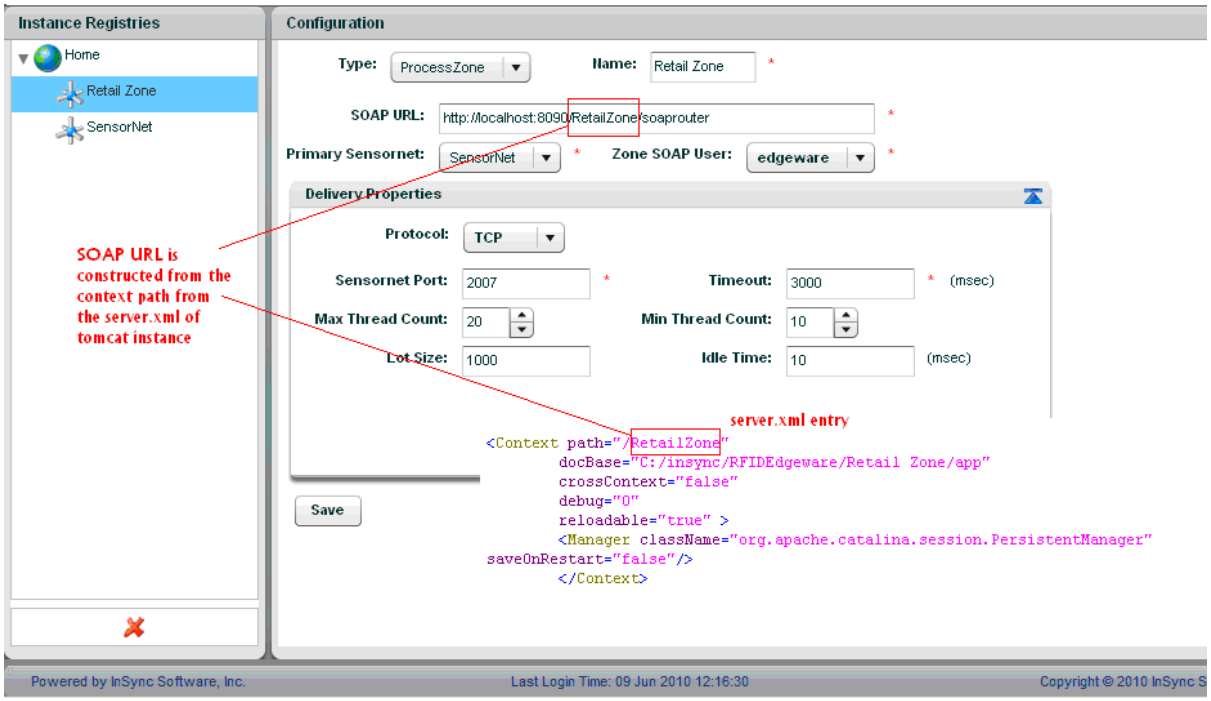
where context path is the value used to construct the url and it can be anything, but should not have any space. Here the url will be http://<machineip>:<portnumber>/RetailZone...

and docBase is the path of the app folder in processzone (Retail Zone) installation.\

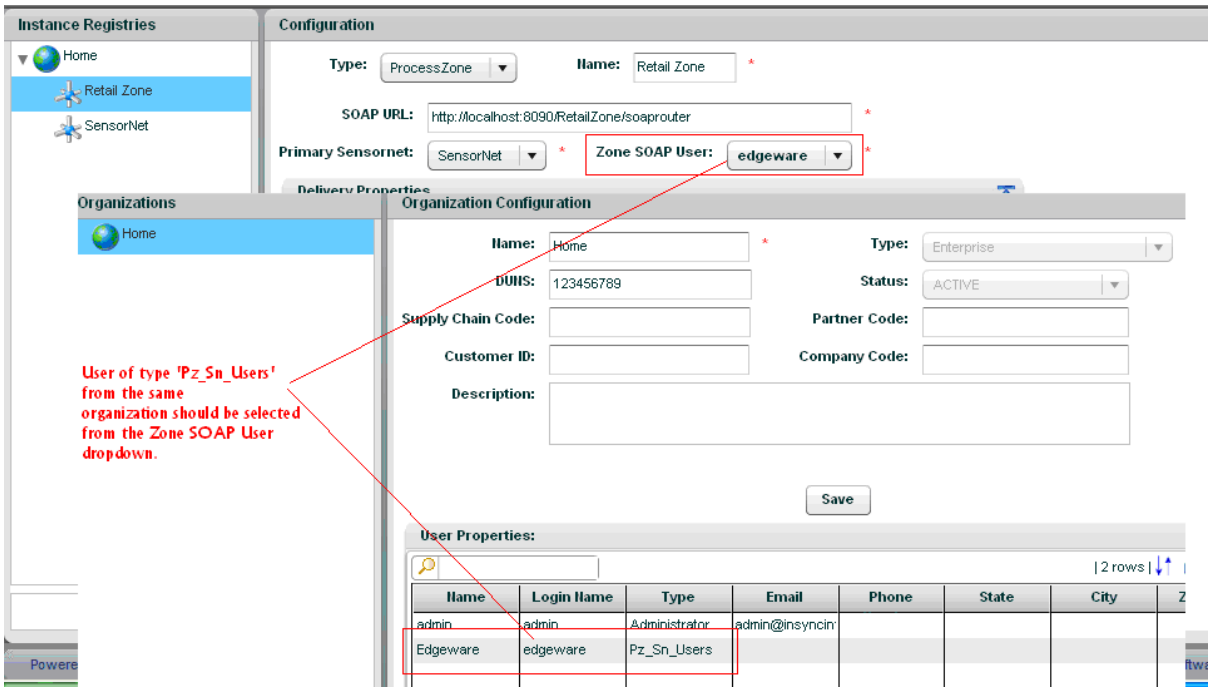
3. If the processzone is installed for an enterprise organization other than the root enterprise organization, then navigate to Administration > Organization screen.
4. Select the organization for which processzone has been installed.
5. Create a user of type 'Pz_Sn_Users'
6. In sensornet, navigate to Administration > Instance Registry screen.
7. Create an instance registry of type 'ProcessZone'.
8. Enter the name exactly same as that given while creating the processzone



9. In ‘ProcessZone’ type of instance registry, in SOAP URL field, edit the machine ip, portnumber and context path of the processzone.



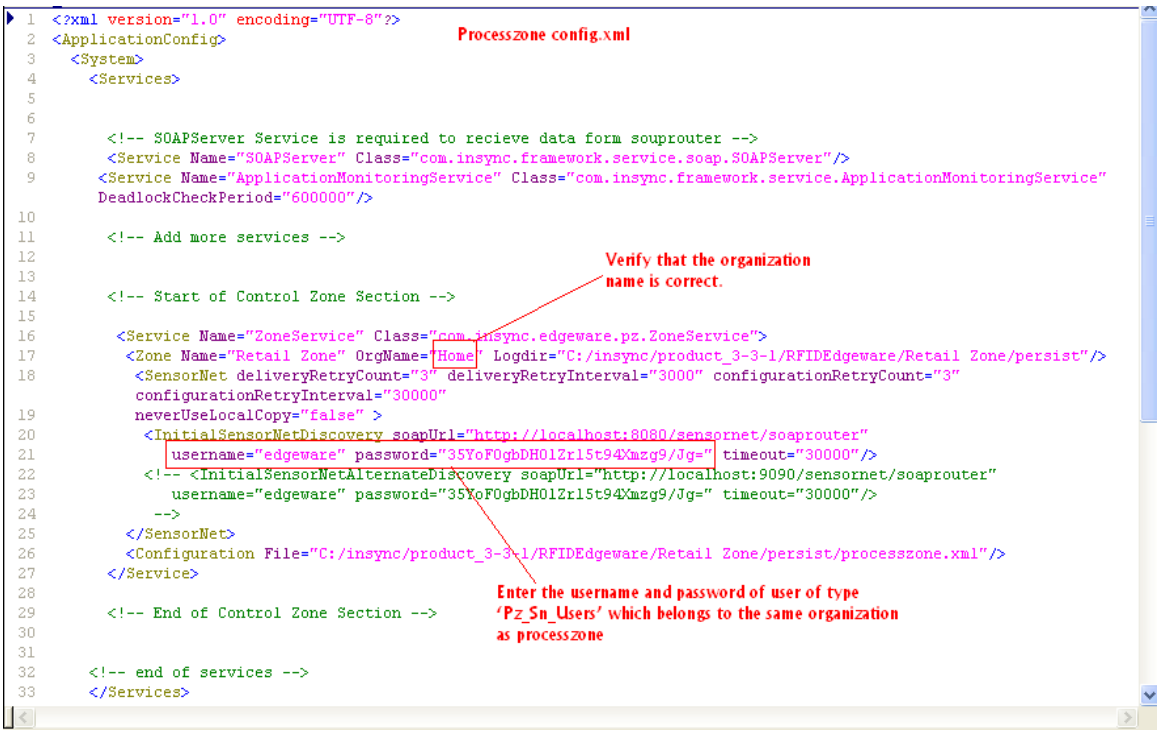
10. Select the user of type ‘pz_sn_user’ from the dropdown for ‘Zone SOAP User’.



11. If there are more than one processzones, Sensornet Port should be different in the instance registries of each of the processzones.
12. Edit the config.xml from '<processzone installation folder>\app\WEB- INF' folder and verify the following:

    i. Zone Name is correct
    ii. OrgName value is correct.
    iii. In InitialSensornetDiscovery, username and password is a user of type 'Pz_Sn_Users' created for the same organization for which processzone has been installed. It can be different from the user selected for Zone SOAP User in step # 22, but both have to be users of type 'Pz_Sn_Users' and should belong to same organization.

Note: When processzone and sensornet are installed on different machines, machine ip should be used instead of 'localhost' in all the configuration files and instance registries.


Processzone config.xml
Verify that the organization name is correct.
Enter the username and password of user of type 'Pz_Sn_Users' which belongs to the same organization as processzone

Process Zone setup sequence

- Create instance registry for the process zone in the Administration->Instance Registry.
- Create zone configuration in the configuration explorer

Follow the above sequence. If you try to do step2 before 1, the screen will not let you do it

Processzone can be installed and running independent of above steps. If processzone is running before step2, then you need to reload the zone from the configuration so that the changes are pushed to the zone. If processzone is NOT running before step2, then no need to reload the zone.

There is no need to restart sensornet and the zone will be sending the reads to sensornet and sensornet would process them.

Limitation

Whenever we modify the instance registry for zone, the changes are not taken into consideration.

Work-around

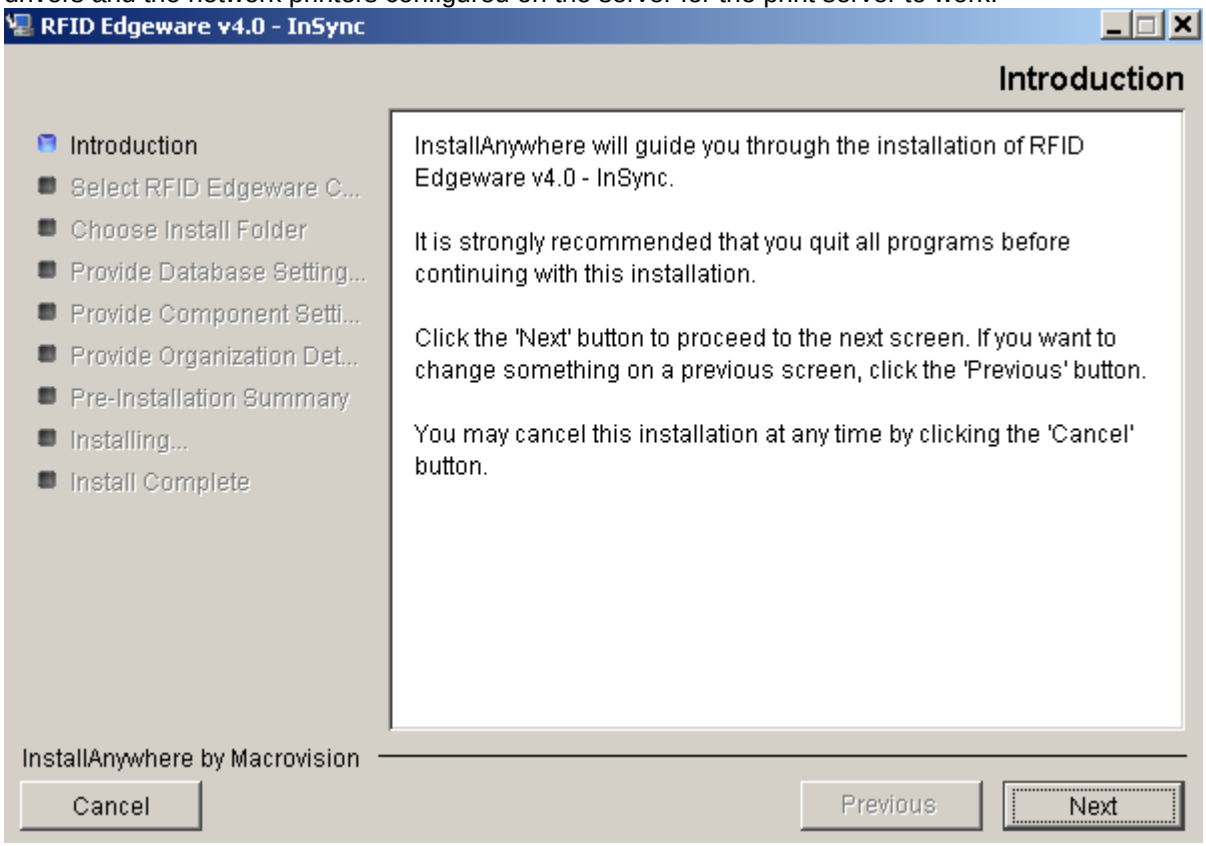
If you make mistakes in step1 and you have NOT performed step2, then you can modify the instance registry , there is no issue
If you make mistakes in step1 and you have completed step2 also, then you need to stop zone tomcat (if running), delete insatnce registry for zone, delete zone configuration and do step1, step2 with correct values.

2.5.6 Installing Print Server

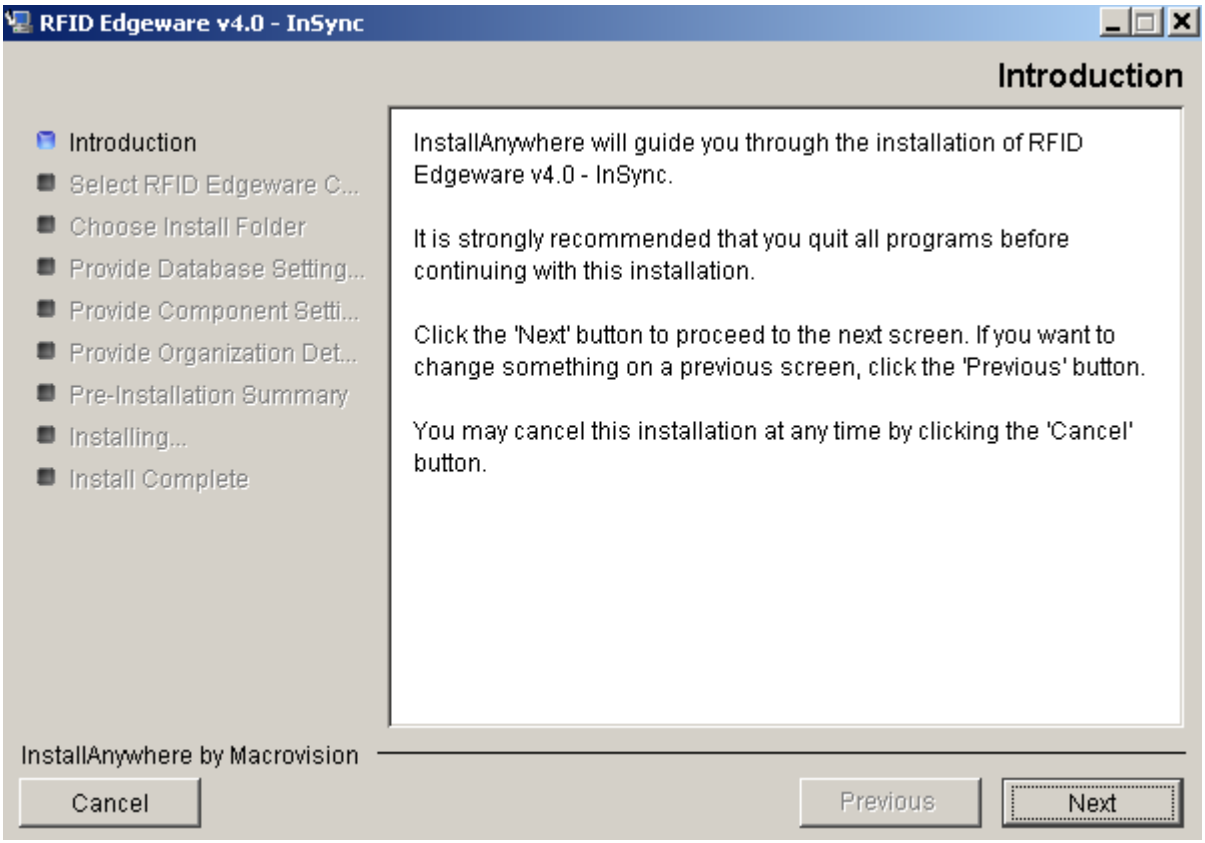
To install print server

1. Double click the install.exe file. This will launch the installation program as shown below.

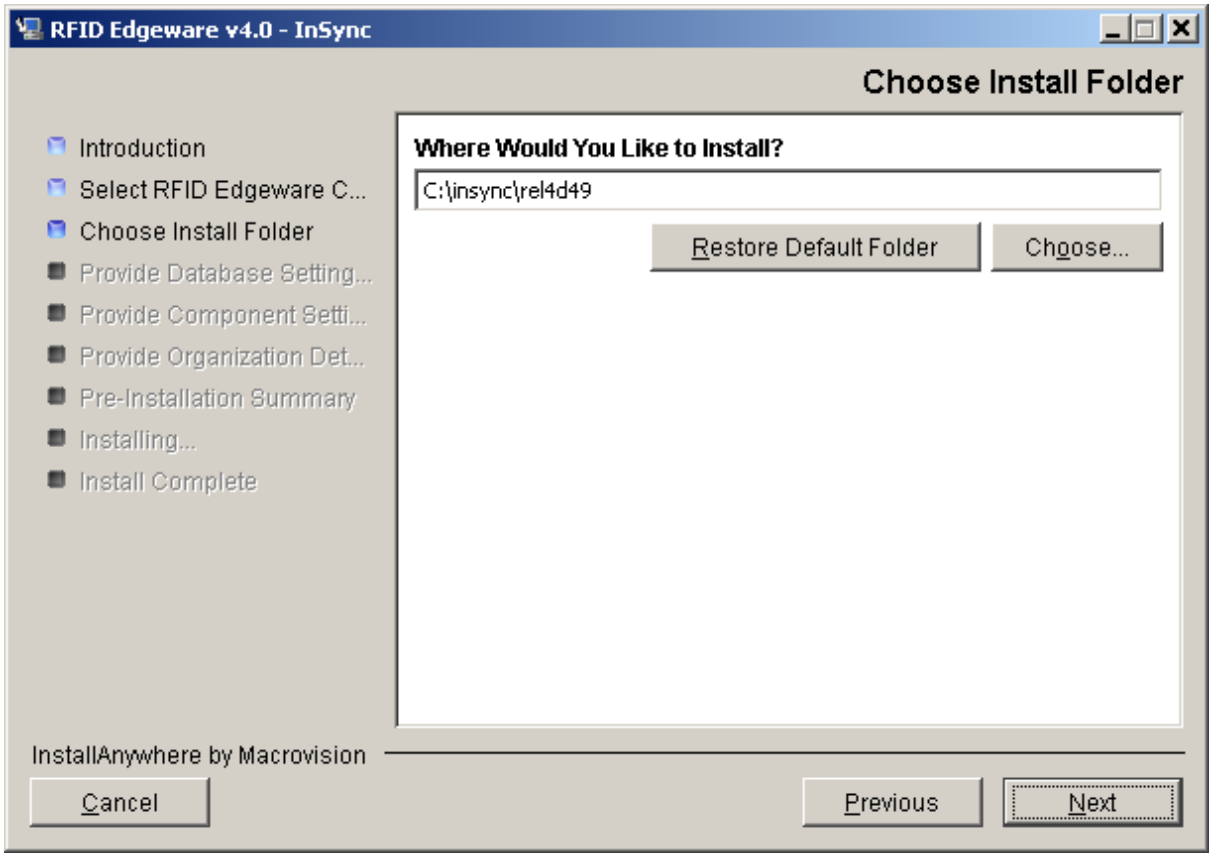
Note: The server on which the print server will be installed would need to have access to the printer drivers and the network printers configured on the server for the print server to work.


RFID Edgeware v4.0 - InSync
Introduction
Introduction
Select RFID Edgeware C...
Choose Install Folder
Provide Database Setting...
Provide Component Setti...
Provide Organization Det...
Pre-Installation Summary
Installing...
Install Complete
InstallAnywhere will guide you through the installation of RFID Edgeware v4.0 - InSync.
It is strongly recommended that you quit all programs before continuing with this installation.
Click the 'Next' button to proceed to the next screen. If you want to change something on a previous screen, click the 'Previous' button.
You may cancel this installation at any time by clicking the 'Cancel' button.
InstallAnywhere by Macrovision
Cancel
Previous
Next

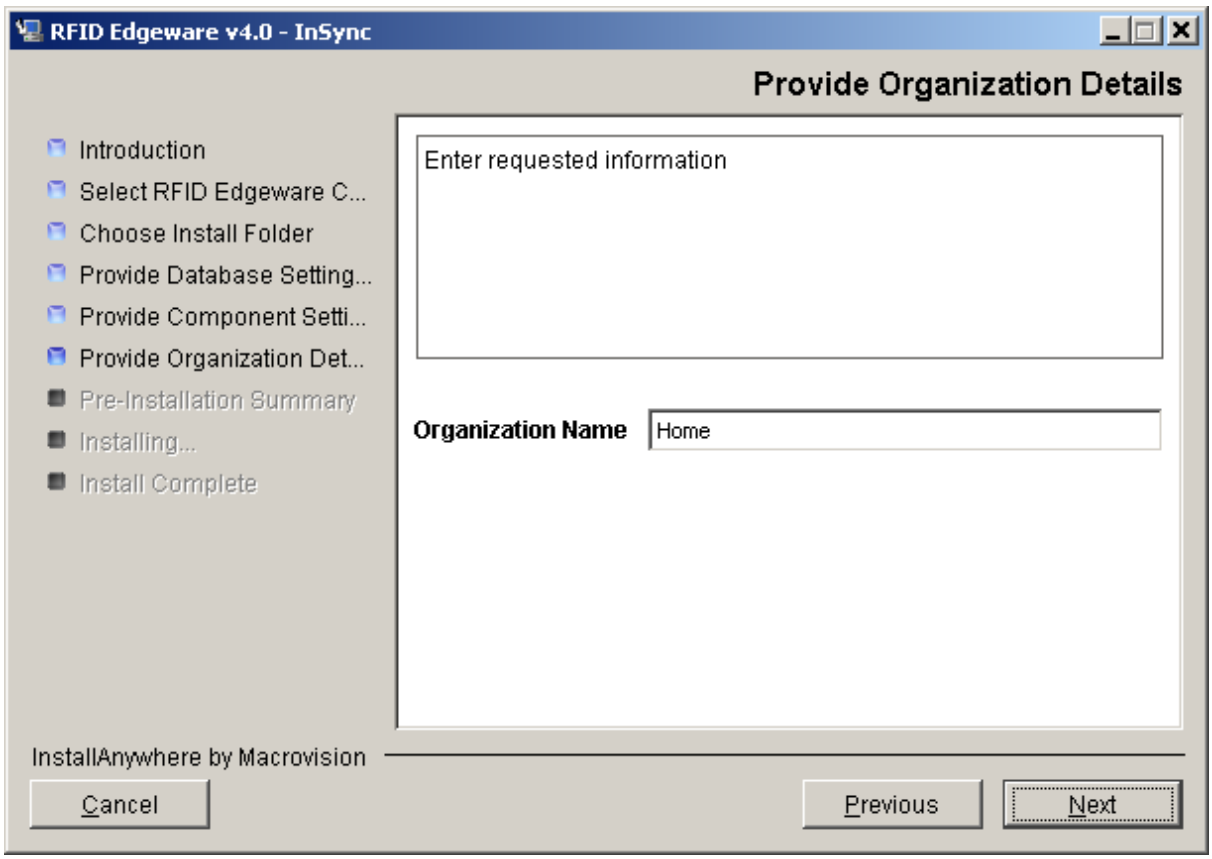
2. Click Next.
3. Select Print server and click Next.


RFID Edgeware v4.0 - InSync
Introduction
Introduction
Select RFID Edgeware C...
Choose Install Folder
Provide Database Setting...
Provide Component Setti...
Provide Organization Det...
Pre-Installation Summary
Installing...
Install Complete
InstallAnywhere will guide you through the installation of RFID Edgeware v4.0 - InSync.
It is strongly recommended that you quit all programs before continuing with this installation.
Click the 'Next' button to proceed to the next screen. If you want to change something on a previous screen, click the 'Previous' button.
You may cancel this installation at any time by clicking the 'Cancel' button.
InstallAnywhere by Macrovision
Cancel
Previous
Next

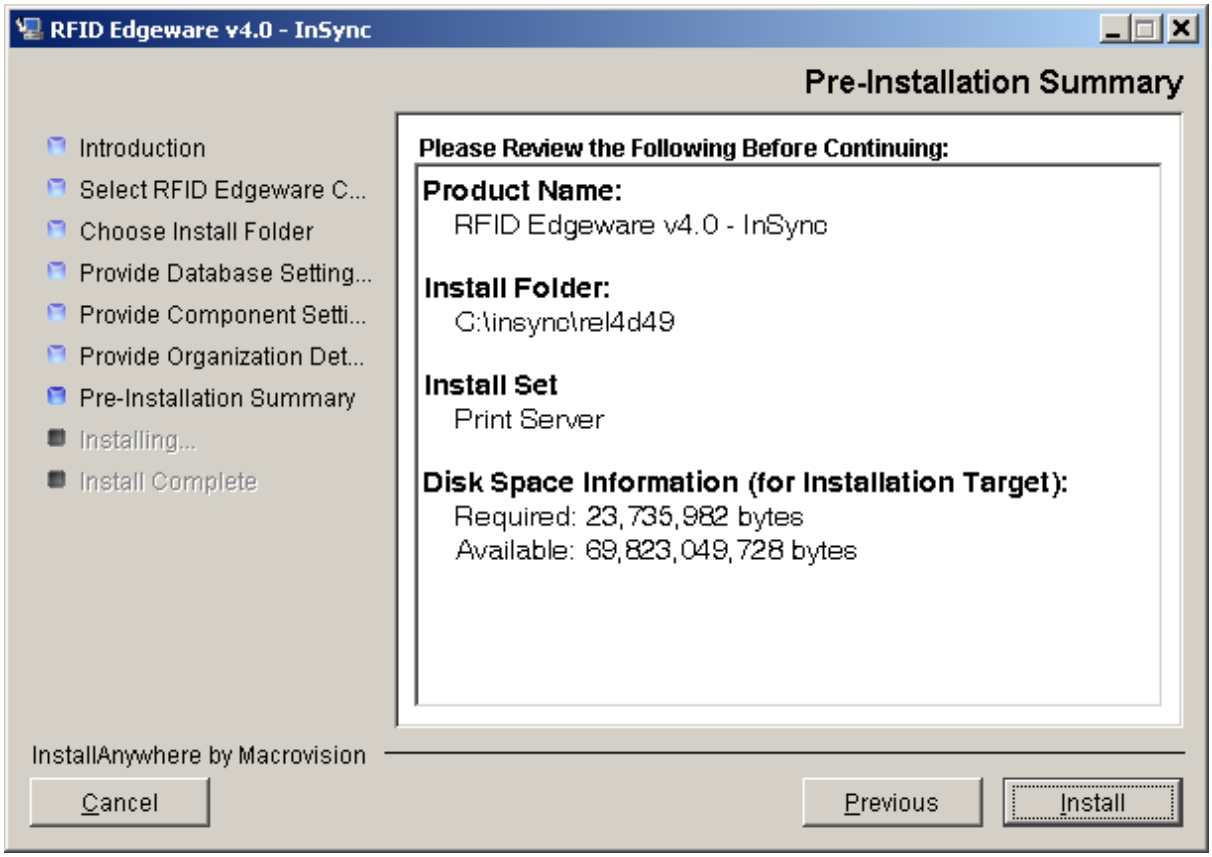
4. You can install the component in the default location or select the destination folder by clicking Choose.



5. Click Next.
6. Enter the Enterprise Organization for which the Print server is getting created and installed. This is the name of the enterprise to which this print server belongs to and which is going to use the application.

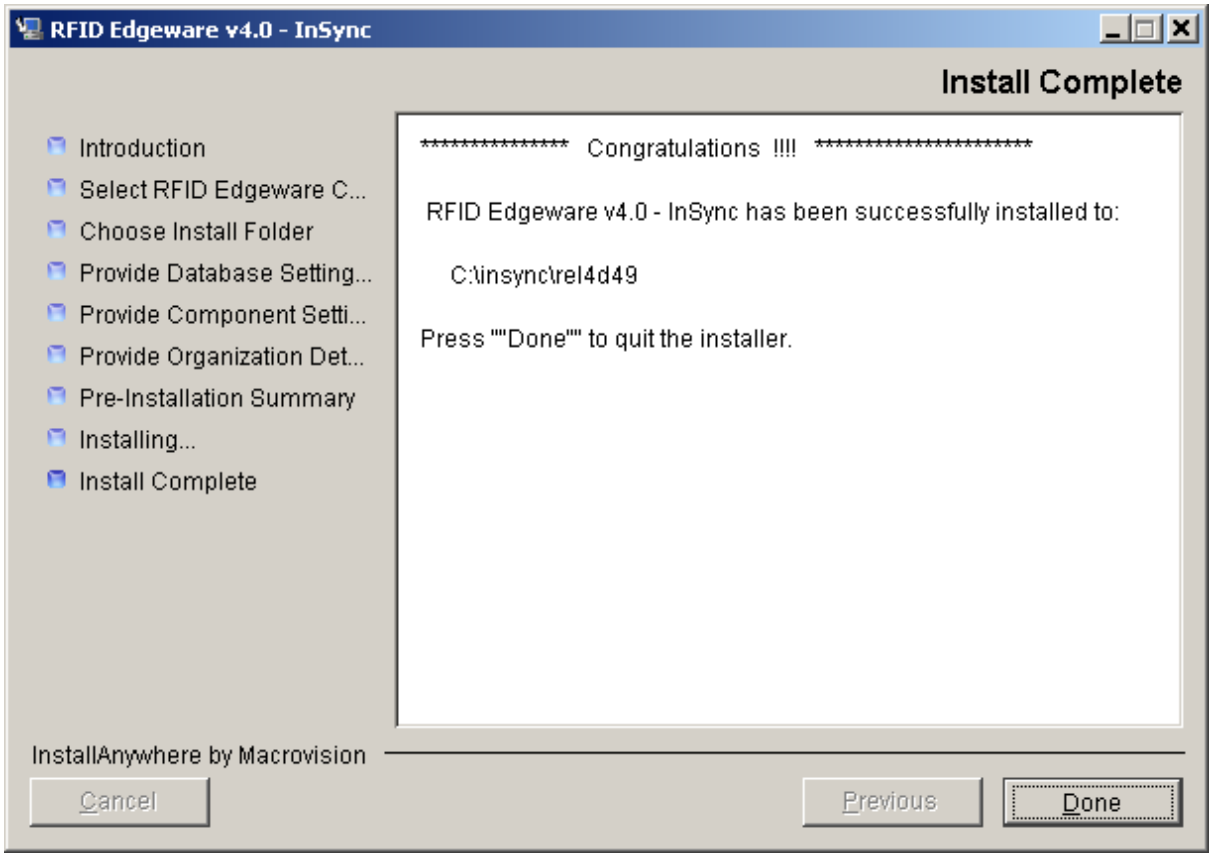


7. Click Next.
8. Now that all configuration details are entered, a confirmation is displayed with all the settings.
9. Use this confirmation to verify the settings.



10. Click Install. Wait for the installation to complete.

11. After successful installation the following screen is shown. Click Done to finish the installation. In case of failure delete the installation folder and check the software versions supported by IApp. The document is available along with this installation document. You will need JDK and not JRE when using the installer.



The RFID IApp™ installation is complete for Process Zone.

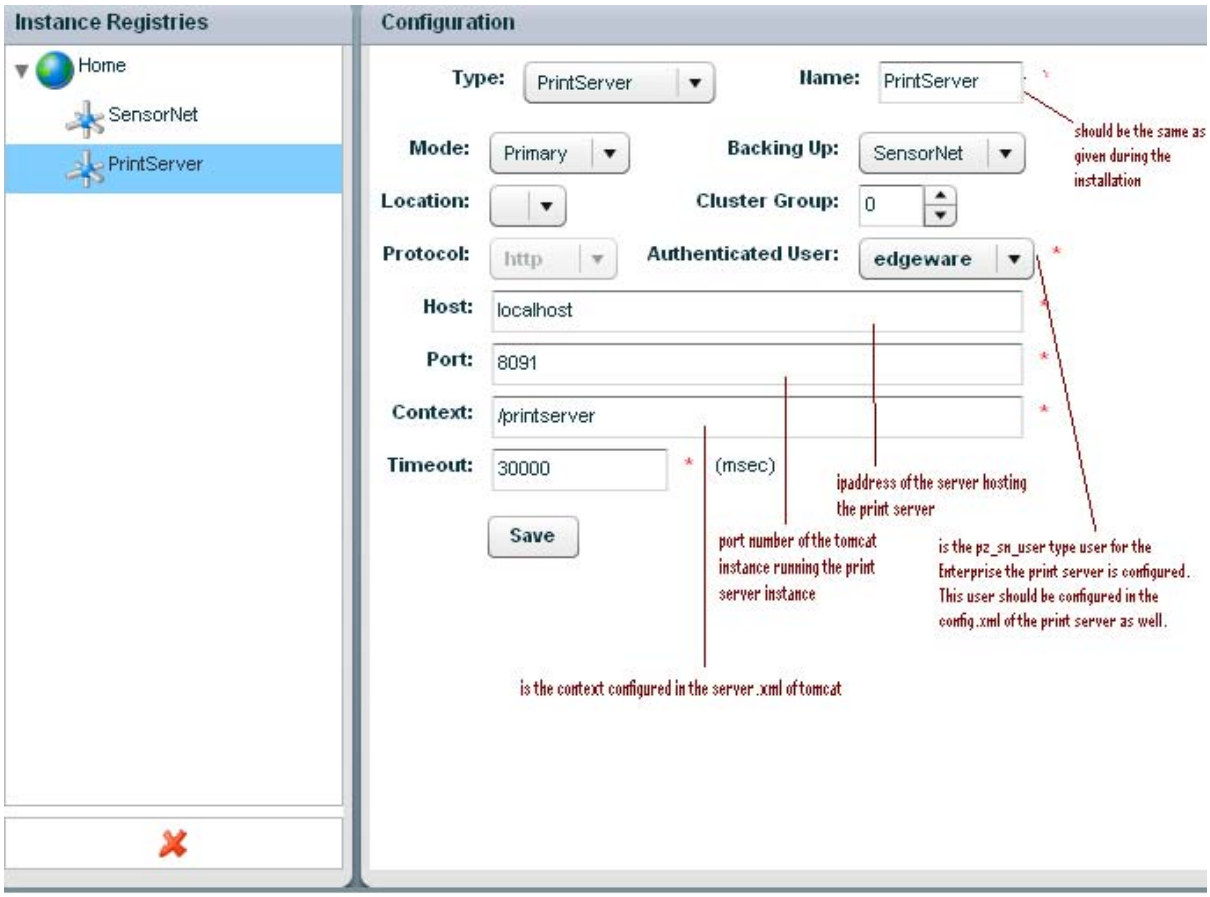
Post Installation steps for Print Server

1. Copy printserver.license.lic to <printserver_install_dir>\app\web-inf\classes folder.
2. Edit server.xml file in tomcat server. Add the following entry:

```
<Context path="/printserver"
                    docBase="c:/insync/RFIDEdgeware/printserver/app"
                    crossContext="false"
                    debug="0"
                    reloadable="false">
            </Context>
```

where context path is the value used to construct the url and it can be anything, but should not have any space. Here the url will be http://<machineip>:<portnumber>/printserver
and docBase is the path of the app folder in print server installation.

3. If the print server is installed for an enterprise organization other than the root enterprise organization, then navigate to Administration > Organization screen.
4. Select the organization for which the print server has been installed.
5. Create a user of type 'Pz_Sn_Users'. This user credentials should be changed in the config.xml located in <printserver_install_dir>/ RFIDEdgeware\printserver\app\WEB-INF
6. In sensornet, navigate to Administration > Instance Registry screen.
7. Create an instance registry of type 'PrintServer'.
8. Enter the Name exactly same as that given while creating the print server



2.6 Post Installation Steps

1. **Licensing** – For Sensornet, Process Zones and Print Server to run you would need licenses files. Please contact your sales contact at InSync for the license files. You will receive sensornet.license.lic for sensornet and processzone.license.lic for process zone.

    a. Copy the sensornet.license.lic to C:/<Deploy_Dir>/RFIDIApp/ sensornet /app/ WEB-INF\classes
    b. Copy the processzone.license.file to C:/<Deploy_Dir>/RFIDIApp/ <ProcessZone> /app/ WEBINF\classes
    c. Copy the printserver.license.file to C:/<Deploy_Dir>/RFIDIApp/ printserver/app/ WEBINF\classes

Where

<Deploy_Dir> is the directory where Sensornet / Process Zone are installed.
<ProcessZone> is the name of the process zone installed.

2. **Process Zone Config.xml:**

a. Edit config.xml located at C:\ <Deploy_Dir> \RFIDIApp\<ProcessZone>\app\WEB-INF to change the user name and password if the Process Zone is installed at Enterprises other than the Home Enterprise. You will find the user name and password attributes for the InitialSensorNetDiscovery soapUrl element.

A default user name and password which is a valid one for Home Enterprise will be set during the installation process.

Also change the OrgName in the Zone element to the organization that the zone is configured to.

Note: The user name and password should be existing users in the Enterprise Organization for which the Zone has been created. To create the user, login to Sensornet->Administration. Follow the administration manual for more information on setting users.

b. Change the ipaddress of the soapUrl=http://localhost:8080/sensornet/soaprouter from the localhost to the ipaddress.

3. **Print Server config.xml**

a. Edit config.xml located at C:\ <Deploy_Dir> \RFIDIApp\<PrintServer>\app\WEB-INF to change the user name and password if the print server is installed at Enterprises other than the Home Enterprise. You will find the user name and password attributes for the Service element.

A default user name and password which is a valid one for Home Enterprise will be set during the installation process.

Also change the organizationname in the service element to the organization that the print server is configured.

Note: The user name and password should be existing users in the Enterprise Organization for which the Zone has been created. To create the user, login to Sensornet->Administration. Follow the administration manual for more information on setting users.

4. **Instance Registry** – It is necessary that the instance registry is setup properly. Login to Sensornet->Administration to be able to set the registry. Refer to Administration section on setting the instance registry.

Note: Every time a Process Zone / Print server is installed, the instance registry has to be setup before they are started.

5. **Sensornet->Configuration->Explorer –** For the Zone to work, it should be created in the Configuration->Explorer page of Sensornet.

Note: Check the zone name for uniformity at the different places it is configured.

- The instance registry name for the process zone should be the same name given at the time of the zone installation.

- The config..xml located at C:\<Deploy_Dir> \RFIDIApp\<ProcessZone>\app\WEB-INF should have the same name again configured at the Zone element as shown below.

<Zone Name="AssetZone" OrgName="Home"

- The zone with the same name as above should be created in the sensornet->Configuration explorer.

2.7 Serial Port Communication

For serial port communication the following files are required.

comm.jar
javax.com.properties
win32com.dll

The above files are available in the iApp installation directory.

- copy comm..jar from <iApp_install_dir>\RFIDEdgeware\<ZoneName>\app\WEB-INF\lib to
- <Java_Install_dir>\lib\ext
- copy javax.com.properties from <iApp_install_dir>\RFIDEdgeware\<ZoneName>\app\WEB-INF\lib to
- <Java_Install_dir>\lib\
- copy win32com.dll from <iApp_install_dir>\RFIDEdgeware\<ZoneName>\app\WEB-INF\lib to C:\WINDOWS\

2.8 Starting RFID iApp™

To setup the Tomcat Steps server for RFID IApp™, you would need to configure it. This is a one time activity. To configure and start Tomcat:

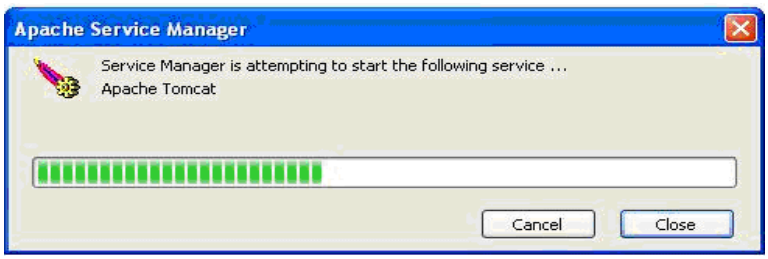
Open “server.xml” file in the C:\Program Files\Apache Software Foundation\Tomcat 5.5\conf folder.
Add following lines in the <Host > section:

```
<Context path="/sensornet"
docBase="c:/insync/RFIDIApp/sensornet/app"
crossContext="false"
debug="0"
reloadable="false" >
</Context>

<Context path="/ProcessZone"
docBase="c:/insync/RFIDIApp/ProcessZone/app"
crossContext="false"
debug="0"
reloadable="false" >
</Context>

<Context path="/printserver"
docBase="c:/insync/RFIDIApp/RFIDEdgeware/printserver/app"
crossContext="false"
debug="0"
reloadable="false">
</Context>
```

Note: *The ProcessZone should be the actual Zone name configured.*

If Sensornet and Process Zones are run on different tomcat instances then it is recommended that Sensornet is started first and then the Process Zones.

Right-click Apache Tomcat service in the Systems Taskbar and select Start Service. You can also start the Tomcat service by browsing X:\Program Files\Apache Software Foundation\Tomcat 5.5\bin



After starting Sensornet tomcat instance, access http://localhost:8080/sensornet to start using the Sensor Net Manager.

The Organization drop down lists all the Enterprise Organizations. The username/password credentials should be valid for the Enterprise selected from the drop down.

***Note**: In the URL given above, you can replace "localhost" with either IP Address or the Machine Name where the component is installed.*

2.9 View Log

When the user is using the application some files are generated and updated on regular basis. These files are known as “Logs”. All events that are triggered using the application are logged into these log files. The users can refer these logs to understand the application’s behavior. In case of any error or when the application stops working the user can view these logs to find out what exactly has happened and take necessary action.

To view all the activities: c:\insync\RFIDIApp\<Component>\logs\edgeware.log.
To diagnose the application:c:\insync\RFIDIApp\<Component>\logs\edgeware_debug.log
The <**component**> could be one of the following:
DefaultZone (or any name you provided during install process)
Sensornet / Print Server

3 Installation Manual - Install Steps on Linux

3.1 Introduction

This document describes the RFID IApp™ Installation program and step-by-step setup instructions on Linux.

3.2 Minimum Hardware and Software Requirements

Hardware

- Intel Xeon/Pentium 4-3Ghz server class.
- 2 Gb RAM minimum.
- 10 Gb free disk space minimum.

Software

- Sun JDK 1.5.0_12
- Jakarta-tomcat-5.0.28
- MySql-server-5.1.45-2

Apache Ant version 1.6.3

Supported operating systems

Red Hat Enterprise Linux 5.0 Server.

Fedora version 13

Other General Requirements

- IE 6.0/7.0
- Macromedia Flash 9
- Screen resolution of 1280 x 1024 pixels.
- To view .XLS reports, MS Excel or Star Office must be installed in the installation system and the viewing system.
- To view .CSV files, MS Excel or another spreadsheet program capable of displaying comma-delimited files must be installed in the installation system and the viewing system.
- To use email alerts, a mail server must be available on the network.

3.3 Pre-Requisites

- JDK 1.5.0_12 installed and is in the classpath. You should be able to do execute the java commands from the shell.
- MySql 5.0 installed and the install folder is in the classpath. You should be able to login to MySql from the shell using the mysql command.
- Ant installed and the installfolder/bin in the classpath. You should be able to run ant from the shell.
- Tomcat installed and able to view http://localhost:8080 in the browser. You should be able to run the tomcat service.

3.4 Pre-Installation Tasks

Checking the Environment Variables

Verify that the environment variables have been set to the values noted in the below table.

Environment Variable	Value
JAVA_HOME	The location of the JDK installation. Note:- To make sure you set your path correctly, run this test with opening a new shell and typing following command: **java -version**
ANT_HOME	The location of the ant installation. Note: - You can check the basic installation with opening a new shell and typing ant. You should get a message like this *Buildfile: build.xml does not exist!* *Build failed*
PATH	Should include the bin directories of the JDK and ANT installations

3.5 Setting the profile for Environment Variables

Edit /etc/profile

Add the following entries (Check for the valid paths)

export JAVA_HOME=/usr/java/jdk1.5.0_12/
export PATH=$PATH:/usr/java/jdk1.5.0_12/bin:$PATH:/usr/local/ant/bin

CLASSPATH=$CLASSPATH:/usr/share/java/mysql-connector-java.jar
export CLASSPATH

PATH=$PATH:/usr/local/ant/bin
export ANT_HOME=/usr/local/ant

3.6 Installing RFID iApp™ Fedora Linux 13 distribution

This guide assumes installing iApp on Fedora Linux 13 distribution. Small adaptations could be required depending on the Linux distribution used.

3.6.1 Installing via Installer

Note: - While using installer makes sure X server service is restarted.

- Set executable permission using chmod command to installer install.bin

  chmod +x install.bin

  Now run install.bin file as follows
  prompt# ./install.bin

- Refer to the iApp Installation section for sensornet , process zone and print server specific instructions

3.6.2 Manual Installation

This installation is not managed by an installer wizard. Therefore, it is required to install components manually, such as the sensornet, Processzone and Printserver:

3.6.3 Sensornet Installation Procedure

After downloading simply type in the following commands:

1 Go to a Linux shell and login as the system administrator root:
su
2 Download products package archive (e.g. sensornet.zip*, processzone.zip and printserver.zip)* and unpack this archive to see Sensornet, processzone and printserver folder.
3 Open the sensornet\install folder
4 Open system.properties file in vi editor and change the database related variables for MySQL and set the paths.

e.g.:-
#
Set each of the variables according to your environment setup.
#

# Organization related variables
edgeware.organization.name = Home

Database related variables for mysql

edgeware.database.host = localhost
edgeware.database.port = 3306
edgeware.database.name = sensornet
edgeware.database.user.name = root
edgeware.database.password = insync

# data base can be one of [mysql,sqlserver,oracle,db2]
edgeware.database.type = mysql

WIP Site properties file
edgeware.wip.site.directory =
/home/InSyncRFID/RFIDEdgeware/sensornet/app/wippools/site.properties

# Log directory
edgeware.log.directory = /home/InSyncRFID/RFIDEdgeware/sensornet/logs
edgeware.temp.directory = /home/InSyncRFID/RFIDEdgeware/sensornet/temp
edgeware.persist.directory = /home/InSyncRFID/RFIDEdgeware/sensornet/persist

# DB Syslog properties:
# The maxrecords is the maximum records to log into the database
edgeware.log.database.application = SensorNet
edgeware.log.database.maxrecords = 1000

5 Set executable permission using chmod command to /sensornet/install/install_all.sh
chmod +x /sensornet/install/install_all.sh

6 Now run install_all.sh file as follows

prompt# ./install_all.sh

3.6.4 ProcessZone Installation Procedure

1. Extract the processzone.zip.
2. All contents will get extracted into a processzone folder.
3. Rename the folder to the zone name that you want to give. For ex zone1.
4. Open the zone1\install folder and edit the system.properties file using vi editor

e.g.:-
#
Set each of the variables according to your environment setup.
#

# Organization related variables
edgeware.organization.name = Home

# Log directory for application messages
edgeware.log.directory = /home/InSyncRFID/RFIDEdgeware/zone1/logs

# Log directory to save device, zone reads, events
edgeware.persist.directory = /home/InSyncRFID/RFIDEdgeware/zone1/persist

# Following three are applicable for PROCESS ZONE installation
edgeware.process.zone.name = zone1
edgeware.sensornet.host.name = localhost
edgeware.sensornet.port.number = 8080
edgeware.sensornet.context = sensornet.

5. Set executable permission using chmod command to /zone1/install/install.sh
   chmod +x /zone1/install/install_all.sh

6. Now run install.sh file as follows
   prompt# ./install.sh

3.6.5 Printserver installation Procedure

1. Open the printserver\install folder and edit the system.properties file using vi editor
   e.g.:-

#
Set each of the variables according to your environment setup.
#

# Organization related variables
edgeware.organization.name = Home

# Log directory for application messages
edgeware.log.directory = /home/InSyncRFID/RFIDEdgeware/printserver/logs

# Log directory to save device, zone reads, events
edgeware.persist.directory = /home/InSyncRFID/RFIDEdgeware/printserver/persist

2. Set executable permission using chmod command to /printserver/install/install.sh
   chmod +x /printserver/install/install_all.sh

3. Now run install.sh file as follows
   prompt# ./install.sh

3.6.6 License Files:

1. Talk to Insync representative for the license files required to run iApp. The license files will be sent to you.
2. After successful installation of each of the components copy sensornet.license.lic to the web-inf/classes folder of sensornet
3. copy processzone.license.lic to the web-inf/classes folder of the process zone that has been installed.
4. copy printserver.license.lic to the web-inf/classes folder of the print server that has been installed.

3.6.7 Configure Tomcat

1. Open server.xml located in <tomcatinstalldir>/conf folder and add the following contexts before the </Host> element.

   For sensornet add the following..

   ```
   <Context path="/sensornet"
   docBase="/home/InSyncRFID/RFIDEdgeware/sensornet/app"
           crossContext="false"
           debug="0"
           reloadable="false">
   </Context>
   ```

   For process zone add the following…

   ```
   <Context path="/zone1"
   docBase="="/home/InSyncRFID/RFIDEdgeware/zone1/app"
   crossContext="false"
   debug="0"
   reloadable="false">
   </Context>
   ```

   For print server add the following…

   ```
   <Context path="/printserver"
   docBase="="/home/InSyncRFID/RFIDEdgeware/printserver/app"
   crossContext="false"
   debug="0"
   reloadable="false">
   </Context>
   ```

3.6.8 Setting Memory Parameters

1. Open setclasspath.sh located in tomcat_install_dir/bin. Add the following to the file.

export JAVA_OPTS=”-Dcom.sun.management.jmxremote

```
-Dcom.sun.management.jmxremote.port=7091
-Dcom.sun.management.jmxremote.authenticate=false
-Dcom.sun.management.jmxremote.ssl=false
export JAVA_OPTS="$JAVA_OPTS –Xmx2048m –Xms2048m -XX:MaxPermSize=1024m"
```

Note: Change the memory values based on the hardware at your disposal.

ii. Section 2
Configuring the iApp Environment

4 iApp Configuration

4.1 Introduction

4.1.1 RFID

Radio Frequency Identification or RFID is a method of identifying unique items using radio waves. RFID technology has emerged in response to the need for a next generation bar code. In the simplest terms, an RFID system consists of a tag (transponder) and a reader (interrogator). The technology of RFID deals with the remote collection of information stored on a tag using radio frequency communications. Information stored on the tag can range from as little as an identification number, to kilobytes of data written to and read from the tag, to dynamic information maintained on the tag, such as temperature histories.

Automatic Identification or Auto-ID is a broad term that covers methods of collecting data and entering it directly into computer systems without human involvement. Technologies traditionally considered part of Auto-ID include bar codes, biometrics, RFID, and voice recognition.

Auto-ID technology provides the means to track any object, anytime, anywhere. The Auto-ID system is based upon the use of low-cost smart tags and readers, and unique object-identification schemes. Auto-ID is achieved by replacing today's UPC barcode labels with inexpensive RFID tags based on tiny slivers of silicon which can be embedded into product packaging, or better yet, into products themselves, although Bar Codes will not go away any time soon. There are many applications where RFID adds complexity and cost, and bar codes are perfectly fine. One will see bar codes and RFID tags used hand-in-hand for many years to come.

An Auto-ID Network comprises various trading partners using the Auto-ID system for tracking and tracing items automatically throughout the supply chain. This provides businesses with an unprecedented real-time view of their assets and inventories anywhere, thereby enabling significant gains to operational efficiencies and brand protection efforts. The Auto-ID Network supplies benefits beyond operational efficiencies by enabling safe and secure supply chains with applications that address counterfeiting, tampering, terrorism, and regulatory compliance, among others.

Refer to the Introduction for more information on RFID fundamentals.

Note: IApp supports the following browsers.

1. Full support on IE 6.0.
2. If Mozilla Firefox is the browser then it is strictly recommended that IApp is accessed from a single browser page.

4.2 Logging In

4.2.1 Login Page

All users are required to be authenticated before using the application. Login screen allows the user to enter their login information so that the system authenticates the user and defines the access control. User can access only those pages that he has permissions.

To access RFID IApp™:

Access a web browser and type http://machinename:portnumber/sensornet

In the URL provided above replace:

REPLACE	WITH
Machinename	Machine name on which RFID IApp™ is running.
Portnumber	Port number on which RFID IApp™ is running (Tomcat). Default is “8080”.

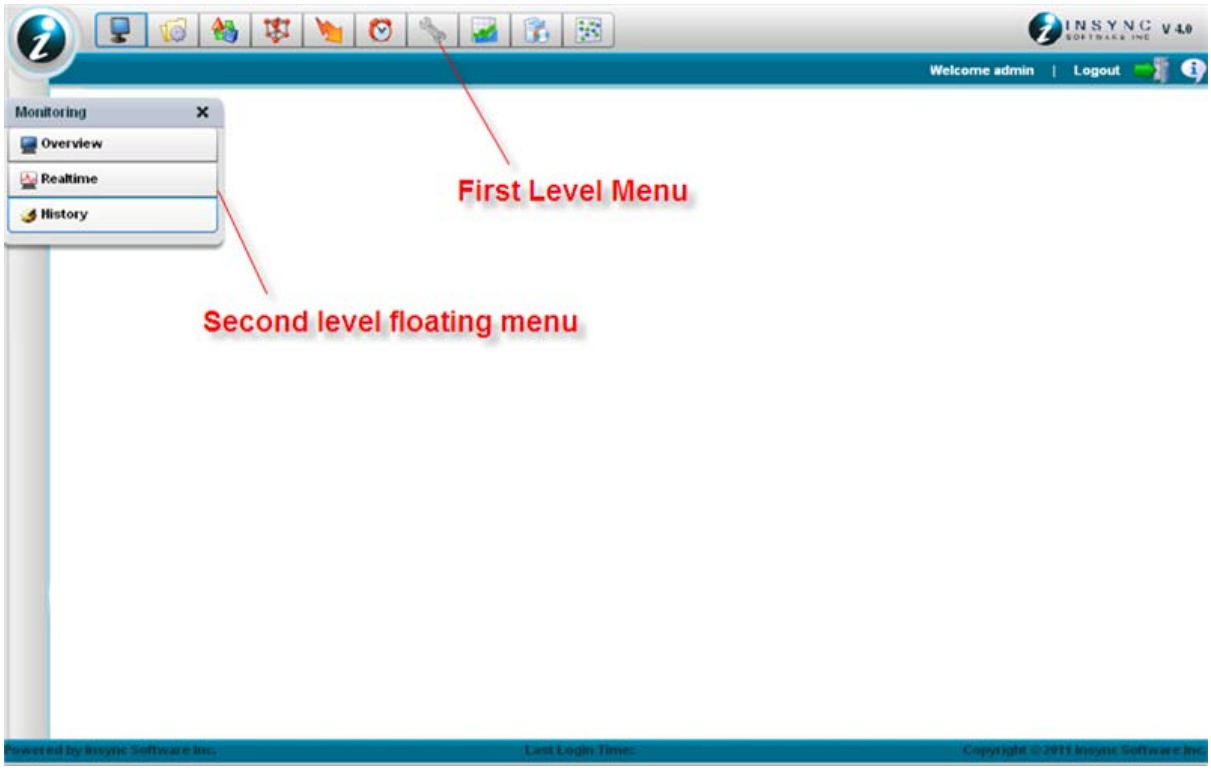
Use the user name and password provided by the administrator of the system. Select the Enterprise Organization from the drop down. Click Ok.

***Note**: Passwords are case-sensitive; ensure CAPS lock on the keyboard is switched off.*

4.3 Configuration Panel Description

4.3.1 Overview

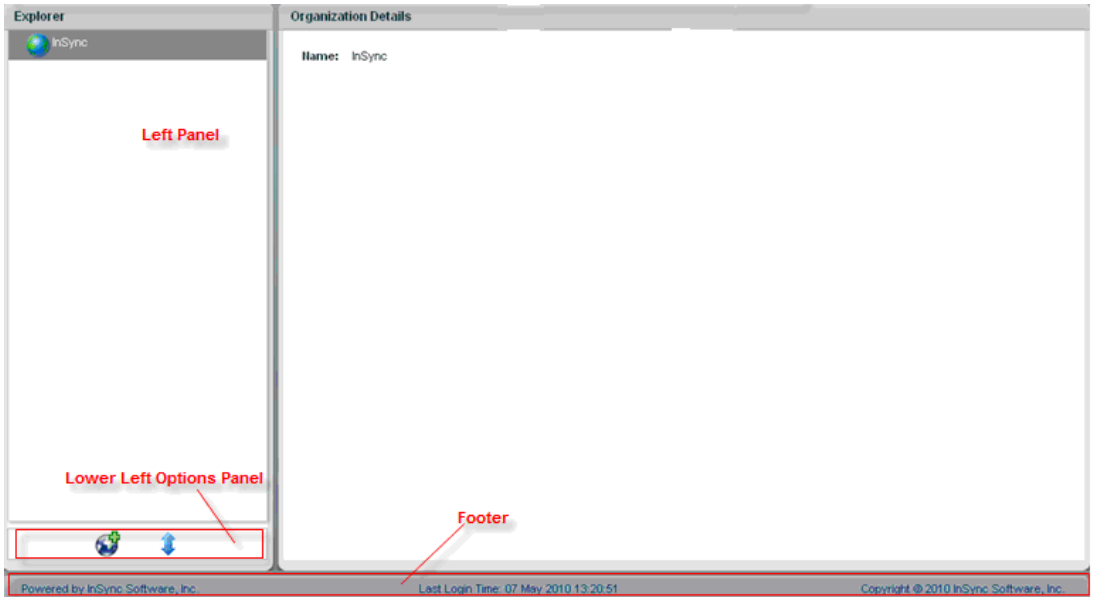
RFID IApp™ has been designed in such a way that once you understand the basics, you'll understand the entire application. There are many interface / usability common elements. These have been outlined below to help you understand how it all works. All screens in the application follow a standard layout as shown below:



Brief description of the interface elements:

ELEMENT	DESCRIPTION
First Level Menu	Allows the user to navigate through the system.
Second Level Floating Menu	Selecting a Top level menu displays the Floating menu options. The menu hides itself on selecting anu option from the menu. Click on the first level menu for it to appear again.

The floating menu lists the items available. The menu gets hidden on selecting any of the options.



ELEMENT	DESCRIPTION
Left Panel	Displays the hierarchy or organization structure in a page.
Right Panel	Displays the information based on the selection in the left panel
Lower Left Options Panel	Displays all the options (Add/Refresh/Export/Import) that are possible based on the selection in the left button panel
Logout	Allows you to log out of the application.
Footer	Displays User name, login time for easy reference.

4.3.2 RFID iApp™ Menu

There are two levels of menus in the application. The Top level menu, as shown in section 4.3.1 points to the major functional components and the second level floating menu shows the specific activities to be performed within each component. The Lower left panel provides all the operations available based on the tree displayed in the left panel. Using the right panel you can create, edit, or delete each record in all functions of the application with few exceptions like reports and monitoring menu items where in you only view reports.

4.3.3 Setting up iApp

Setting up IApp involves using the instance registry under Administration to define all Sensornet instances and the process zones running within each Sensornet. Metadata of the Sensornet and process zones are captured using the feature.

For more information on how to use the instance registry refer to the Administration ->Instance Registry section.

4.3.4 Enterprise Organization

All the views in Sensornet are by Enterprise Organization. It is required that the organization structure is created before proceeding with the configuration of the system. This can be done from the Administration first level menu item.

For more information on how to setup the organization structure please refer to Administration ->Organization section.

Note: At the time of Sensornet installation the system captures the Home (Managed Enterprise) organization name.

4.3.5 Enterprise User Access

Each enterprise organization has its own access rules set using the Administration tab. User access to the Enterprise organization is based on the Roles and Permissions set for the user by the Administrator of the system.

Users belonging to the Home Enterprise can be given access to other Enterprise organizations created within a Home Enterprise. When such user logs into IApp then he will see the Organization information of both the Enterprises he is given access to provided that necessary permissions are set. However use can perform transactions for the Enterprise he belongs to but not in the Enterprises he has access to. Also you can configure and view for Enterprises he has access to.

Users belonging to Enterprise organizations other than the HOME organization will have access to only that Enterprise information only.

For more information on setting up users and permissions please refer to Administration ->Users section.

4.3.6 SYSTEM vs. CUSTOM

All objects defined/created in IApp are grouped as SYSTEM or CUSTOM based on whether the object has significance across all Enterprises or for a particular Enterprise. Objects which are used across all Enterprises are viewed/created as SYSTEM and objects which are of significance to only a particular Enterprise are created as CUSTOM within that Enterprise.

Note that though the SYSTEM information is viewed across all Enterprises, the information can be edited by the administrator at the HOME Enterprise only. Also, SYSTEM objects can be created only at the HOME Enterprise and by the administrator only.

Note: The SYSTEM objects are necessary for the smooth functioning of IApp. It is recommended that the administrator do not delete or tamper the information without proper guidance.

4.3.7 RFID iApp Sensornet

Sensornet plays a crucial role of controlling, managing, and monitoring multiple Enterprises and the Process Zones created at each Enterprise. At a Zone level it controls, manages and monitors RFID locations and devices.

Each Sensornet instance is capable of managing multiple Enterprises. The following sections will explain the different components of Sensorent.

Note: For purpose of documentation there are 2 Enterprises created – InSync as the HOME Enterprise and ABC Corporation as the Enterprise created within the HOME Enterprise. The user logged in is the HOME Enterprise user with access to ABC Corporation.

Note: Entity Modeler and Dashboard Menu items are not covered in this version of the document.

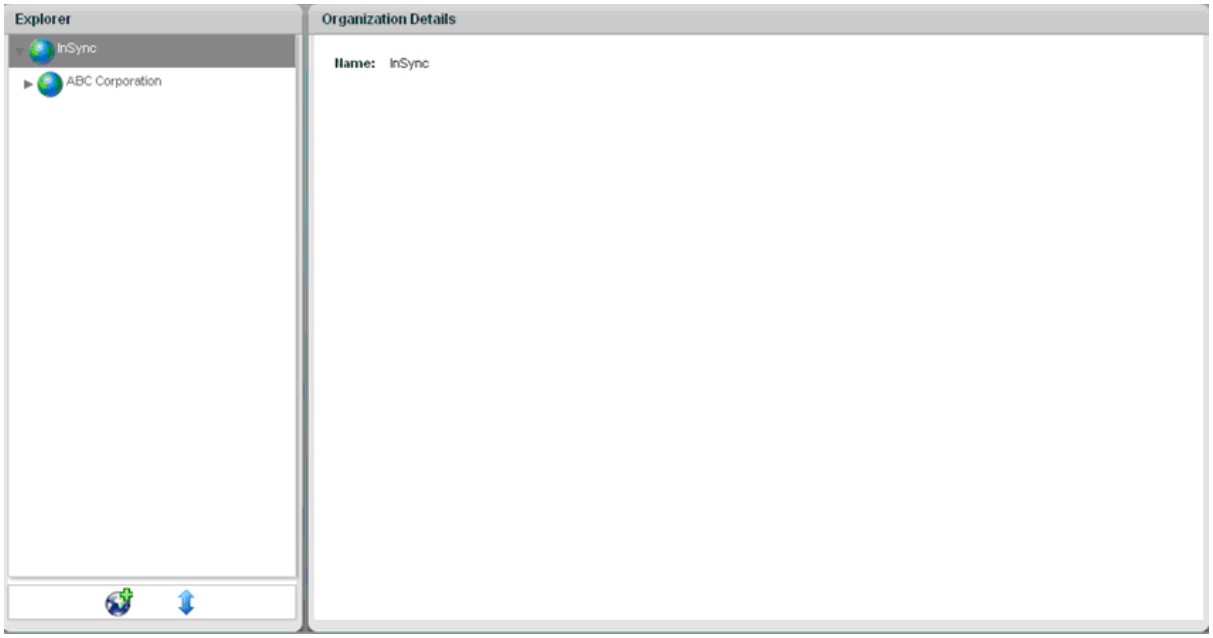
4.4 Configuration

Configuration in Sensornet allows for setting up the Zones, Locations, Devices, Tag Schemes and Edge Processes by Enterprise Organization. Configuration is the starting point of setting up the application configuration based on the use case.

Note: Before you start of on the configuration make sure that the Organization structure is in place and users setup with all the required access. For more information check the RFID IApp Installation section.

4.4.1 Explorer

Click in the First Level menu. Select Explorer from the floating menu.



Configuration->Explorer provides the option of creating the locations, process zones, devices and edge processes based on where you are in the Configuration tree displayed in the left side panel. There are certain constraints set by the system for effective navigation while creating this tree.

- Locations can have multiple locations created
- Locations can have multiple Process Zones
- A Process Zone cannot exist in a Zone hierarchy which means that only one process zone can exist in its hierarchy.
- A Process Zone can have its own location hierarchy within it.
- A Device or a Read Point (logical location) can exist in a hierarchy which has a process zone configured higher up in the same hierarchy.
- Locations cannot be created once a Device or a Read Point is created in a hierarchy.

Note: 1. Each Enterprise Organization can have its own configuration setup created from the overview page.

2. - Delete Operation on any node in the Configuration tree is possible till it does not have any child nodes. You will have to delete the child nodes before proceeding to delete the parent node.

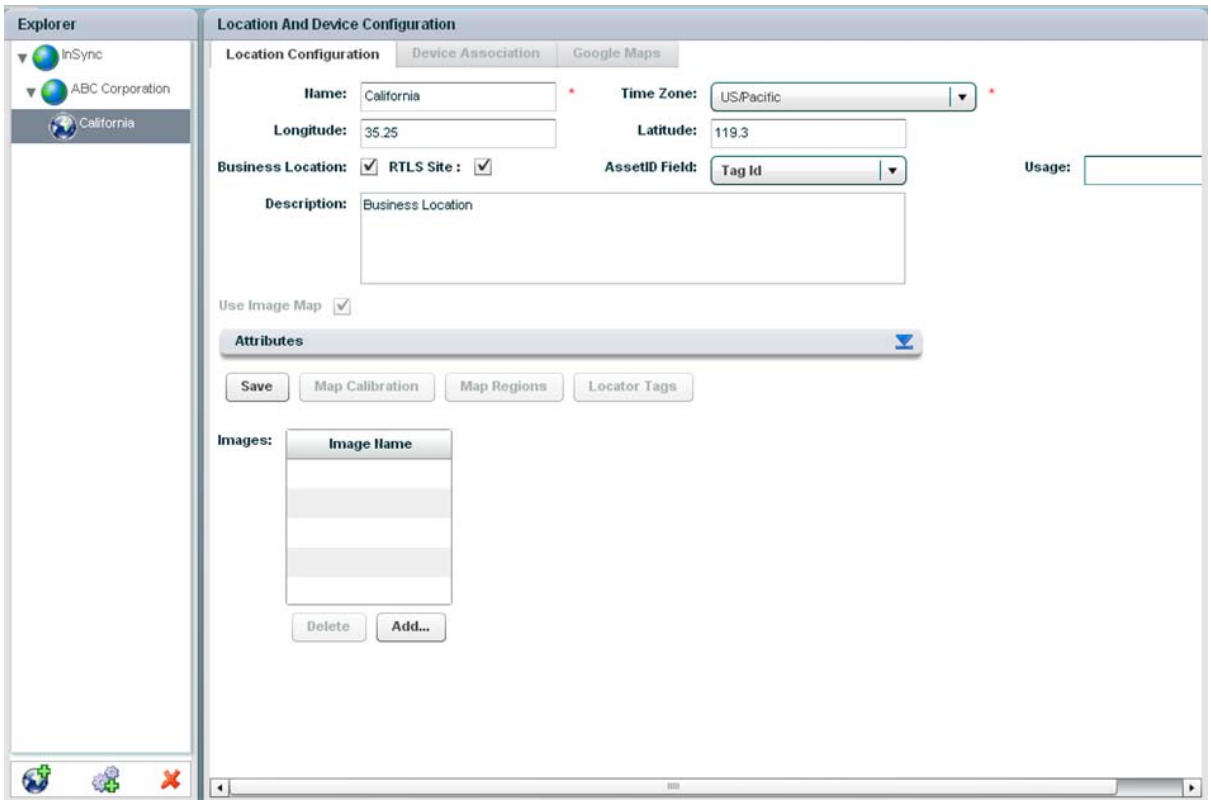
Select the Enterprise Organization in the left panel for which the configuration needs to be created.

Operations possible on the Enterprise Organization are:

- Add a Location
- Import/Export

4.4.2 Location

Select the Organization and Click on to create a Location



Location has the following attributes.

Attribute	Description
Name	Name of the location
Time Zone	Time Zone of the location that is getting created
Longitude	Longitude of the location
Latitude	Latitude of the location
Business Location	Suggests if the location getting created is a Business Location or not.

	Note: Atleast one Business location is required in each Enterprise Organization. A business location cannot be created within a business location. The business locations are used by the business documents, material locations and material traces. All trackables (item, tag, container and history) will be updated with business location at creation and commissioning time.
RTLS site	Suggests if the location getting created is a RTLS site or not.
AssetID Field	The attribute to be considered for the AssetID field. Note: This is valid only for RTLS application and the filed is enabled only if the RTLS site checkbox is checked.
Usage Field	Additional information of the location. This can be used to suggest to the application using the location for a specific purpose.
Description	Brief description
Attributes	These are 10 String and 10 Numeric attributes. Any additional information of the location can be stored using these attributes

- An Image can be uploaded for the location using the Add Image button. Multiple images can be uploaded for a single Location.
- To Edit the details select the Location, make the changes and click Save.
- To delete the location select the Location and click button in the lower left panel.
- Once a Location is created multiple Locations can be created or a Process Zone can be created at the Location.
- A Business location cannot be created within a business location.

Note: The Map Calibration, Map Regions and Locator Tags are for Image Calibration and will be dealt at in a separate section

Device Association:
The device association tab will be enabled if an image exists for the location and is calibrated. If the image exists and is not calibrated then the tab is disabled.

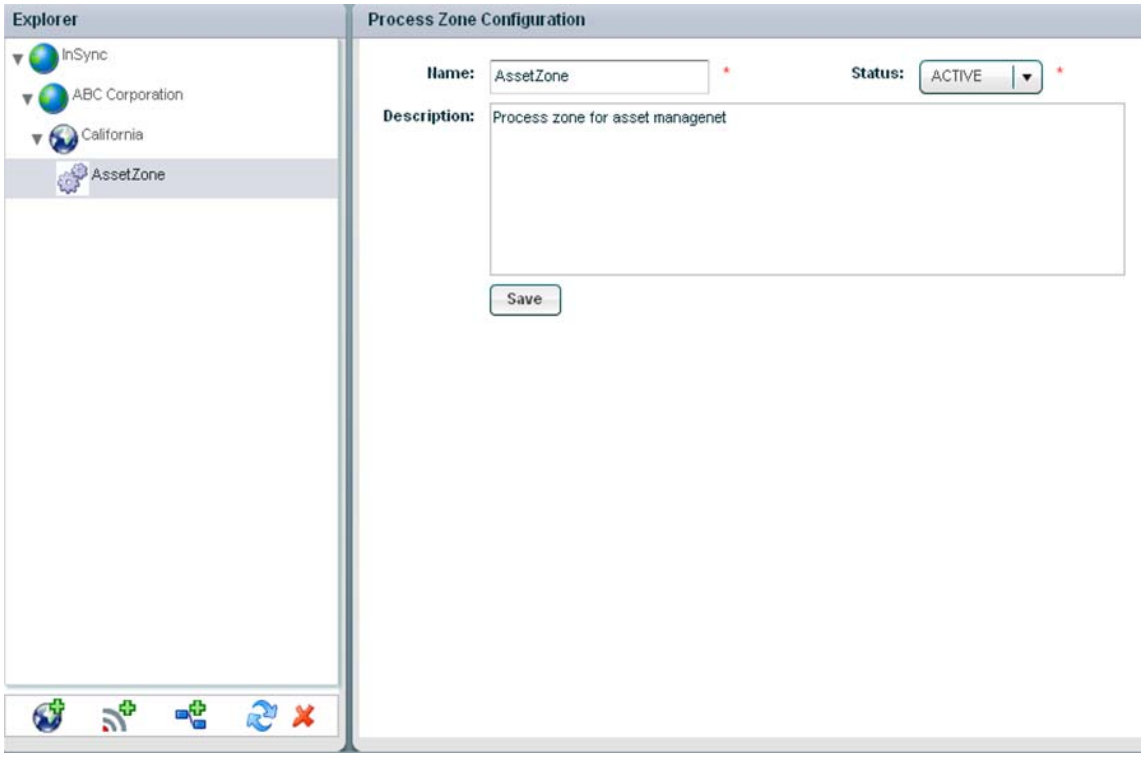
The purpose of the device association tab is to be able to place the devices at the required co-ordinates on the calibrated image.

Note: Google Maps tab and Use Image Map checkbox are disabled in this version.

4.4.3 Process Zone

Process Zone in Sensornet manages all the Devices and Edge Processes. A zone can contain multiple devices; however a device can connect to only one zone. Since the Zone doesn't have a GUI, Sensornet maintains the data about the Zones and the Devices connected to it.

To create a Zone select the Location and click on in the lower left side panel.



Note: The Zone name should be the same as the one given for the zone at the time of installation. It is advised that the Zone is installed using the installer before it is created. As a pre-requisite to create the zone the zone should be setup in the instance registry.

Zone name is unique across all organizations. The zone name once it is created cannot be modified again.

Process Zone setup sequence

- Step 1: Create instance registry for the process zone in the Administration->Instance Registry.
- Step2: Create zone configuration in the configuration explorer

Follow the above sequence. If you try to do step2 before 1, the screen will not let you do it

Processzone can be installed and running independent of the above steps. If processzone is running before step2, then you need to reload the zone from the configuration so that the changes are pushed to the zone. If processzone is NOT running before step2, then no need to reload the zone.

There is no need to restart sensornet and the zone will be sending the reads to sensornet and sensornet would process them.

Limitation

Whenever we modify the instance registry for zone, the changes are not taken into consideration.

Work-around

If there is a mistake in step1 and step2 has still not be performed, then the instance registry can be modified without any issue.
If there is a mistake in step1 and step2 has been performed, then stop the zone tomcat (if running), delete instance registry for zone, delete the zone configuration and do step1, step2 with correct values.

Operations

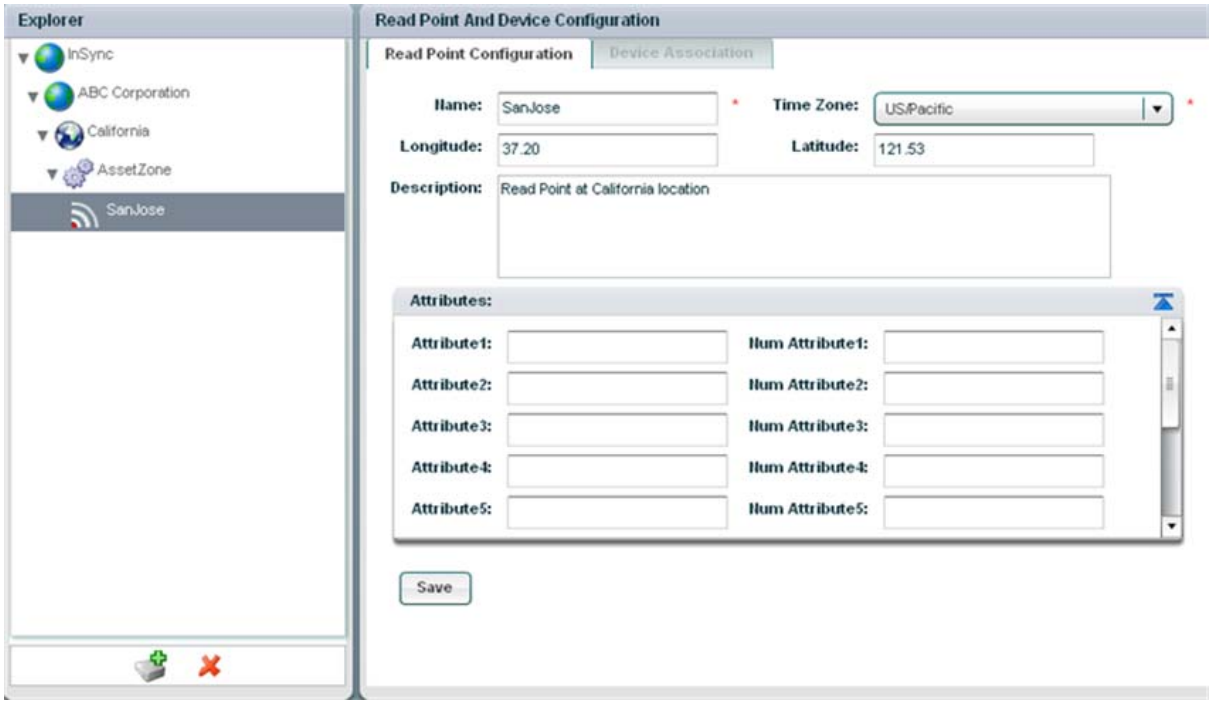
The operations that are possible on the zone are.

- Add a Location
- Add a Read Point – Read Point is of type Location except that once a Read Point is created at a location only a device can be added and further location configuration cannot continue beyond the read point.
- Add Edge Process
- Reload Process Zone – If any of the configuration changes in the Zone hierarchy then the Zone should be reloaded for the changes to get affected.
- Delete the Process Zone

4.4.4 Read Point

Read Point is the location that has a reader installed.

Select the Process Zone or a Location within the Zone and Click on to add a Read Point.



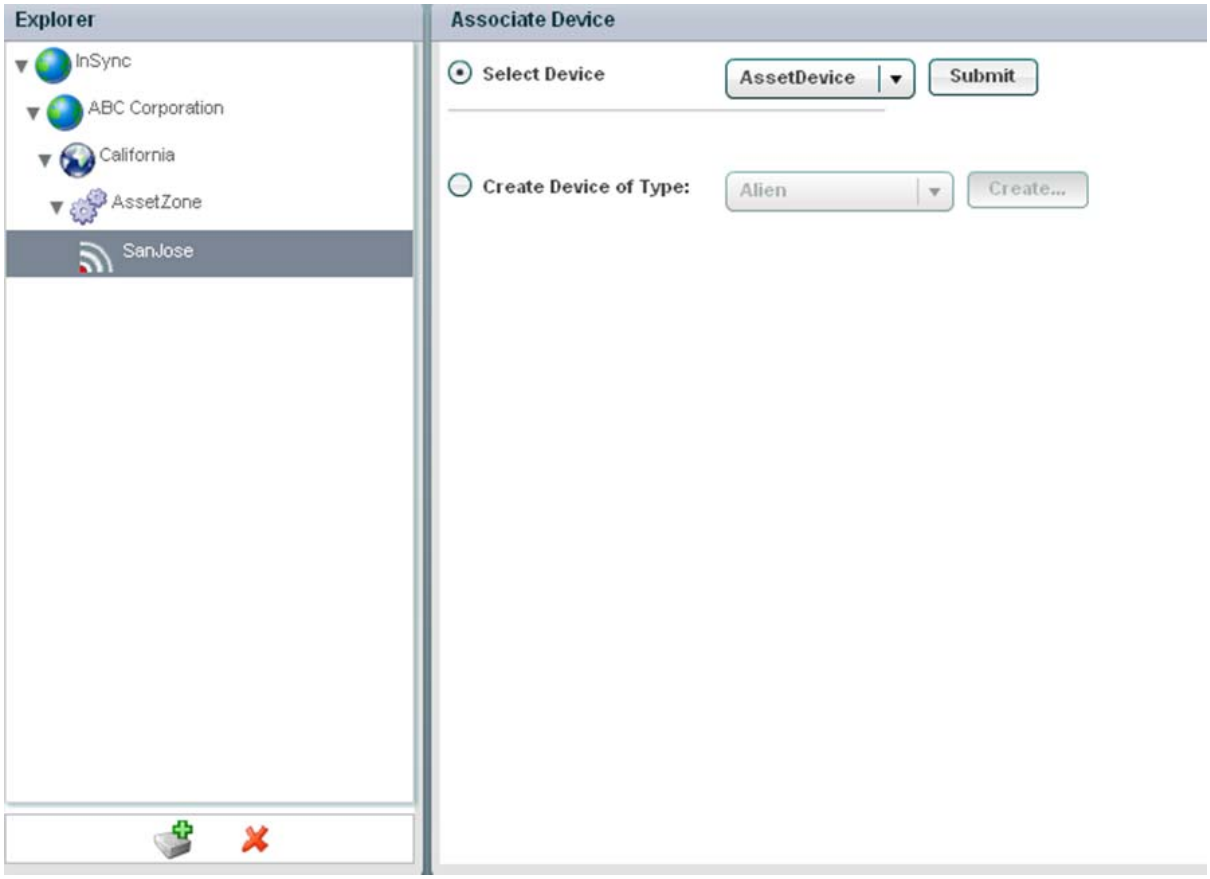
Read Point has the following attributes.

Attribute	Description
Name	Name of the read point. The name is unique within an Enterprise organization.
Time Zone	Time Zone of the read point
Longitude	Longitude of the read point
Latitude	Latitude of the read point
Description	Brief description
Attributes	These are 10 String and 10 Numeric attributes. Any additional information of the read point can be stored using these attributes

- To Edit the Read Point select the Read Point, make the changes and click Save.
- To delete the Read Point select the Read Point and click button in the lower left panel.
- Once a Read Point is created, creation of Locations is not possible in the same tree which means that the Location hierarchy ends at the creation of a Read Point
- Operations possible on a Read Point are
  - Add a Device
  - Delete the Read Point

4.4.5 Add Device

Select the Read Point and click on to Add a Device to the Read Point.



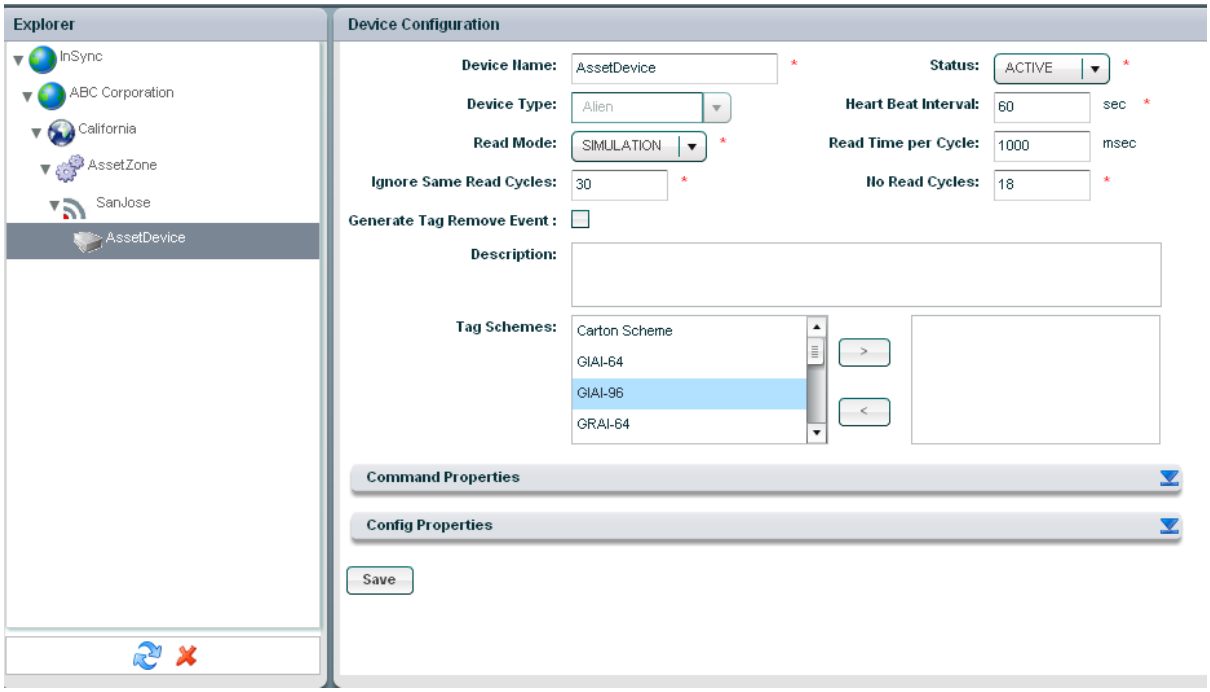
There are 2 options that are possible while adding a device.

- Option 1- Select a Device from the list of Devices already created and not associated it to the Read Point. This set of devices is created in the Devices Tab which is explained in later sections.
- Option 2- Create a new device by selecting the Device Type from the drop down.

Note: Adding new Device Types is an administrative task and is accomplished from the software tab accessible only to the Home enterprise administrative users.

Option 1:

- Check the radio button against ‘Select Device’.
- Select the device from the drop down. Devices which are not associated to any of the Read Points will be visible here.
- Click Submit.
- The device gets added to the Location tree in the left side panel.
- Enter the details and click ‘Save’ at the end of the page.



The different attributes for the Device are

Attribute	Description
Name	Name of the device. The Device name is unique across all enterprises. Device with the same name cannot exist in more than one enterprise organization.
Status	Denotes whether the Device is Active or Inactive
Device Type	Type of Device being configured
Heart Beat Interval	Interval at which the system monitors the health of the device
Read Mode	Mode to which the device is tuned. • Auto mode polls the Reader for reads continuously until the Reader is turned off or the application is shutdown • Semi-Auto needs the command to be sent in Manual mode at least once before it goes into Auto mode • Manual mode needs the command to be explicitly sent to the Reader; this can be done in Activities tab discussed later. • Simulation mode imitates the Reader's execution of commands; this is useful in development and test environments. **Note:** For Gen2 commands to work, the device has to be in **Manual** Mode ONLY.
Read Time per Cycle	It is the period in milliseconds in which the reads

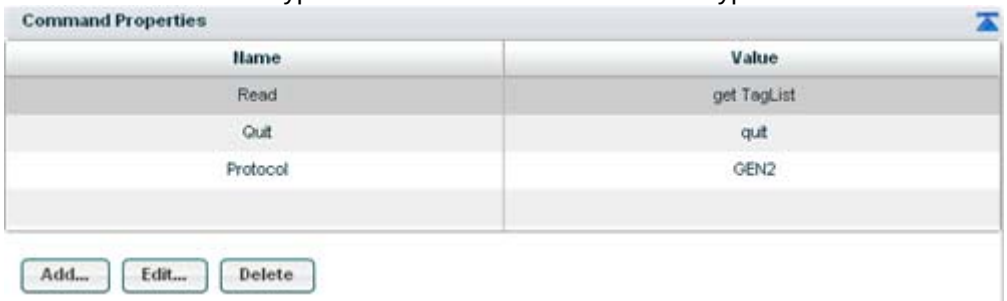
	are read. For ex; if set to 2000ms then the read cycle is 2 seconds meaning the reads are read every 2 seconds
Ignore Same Read Cycles	This is the number of cycles within which the read will be ignored/filtered if it appears again within the cycle from the first cycle it is read. For ex; if set to 10, then the reader will ignore the read if it appears more than once in the 10 cycles starting from the first cycle it is read. Beyond the 10 cycles it will be considered as a new read.
No Read Cycles	If a tag is read in a cycle and no reads happen for the cycles mentioned from the first cycle it was read then the reader considers it to have gone out of its sight and generates a Tag remove event.
Generate Tag Remove Event	If checked then the event is raised based on the 'No Read Cycle' time configured else ignored.
Tag Scheme	Defines the encoding scheme of tag value fields. Select the desired tag scheme from the list box. For more information on tag Schemes refer to the section on Tag Schemes.

Command Properties:

These are device commands along with the values in the form of name value pairs. The names could be variables that are defined in the Device Type inside [] brackets.

Use the min max button in the Command Properties panel to view the command properties. The panel provides means of adding, editing or deleting a command property.

For more information on Device Types refer to the section on Device Types.



Protocol is specified as 'GEN2' by default. You can change the value if you want to execute commands in a Protocol suitable for your device. Most Readers support multiple protocols; valid protocols are ISO18000-6B, EPC0, EPC1, and GEN2.

Config Properties:

Config properties are used to configure and fine tune the device to run properly. Property names are already populated based on the Device Type.

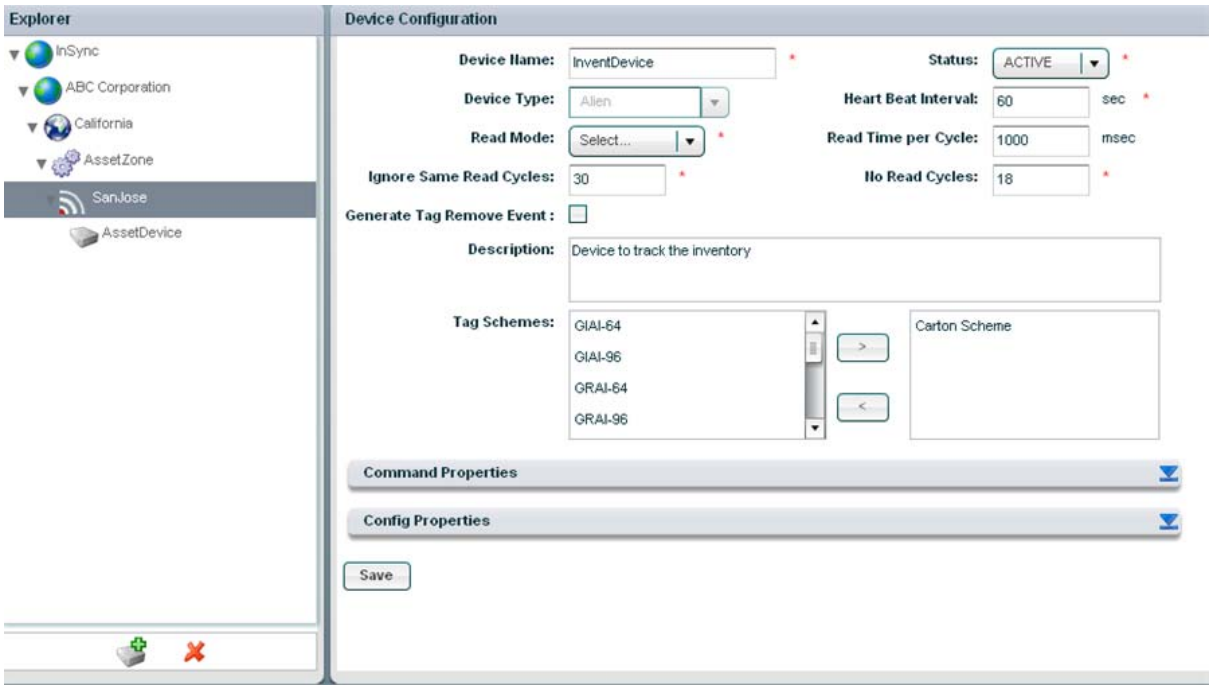
Use the min max button in the Config Properties panel to view the Config properties.
The panel provides means of adding, editing or deleting a command property.

Common Properties of the device- IP address, Antenna count, Send Read Event, Use Device Time, Port Number are configured here.

For more information on the Config Properties for each Device Type refer to Device Types section

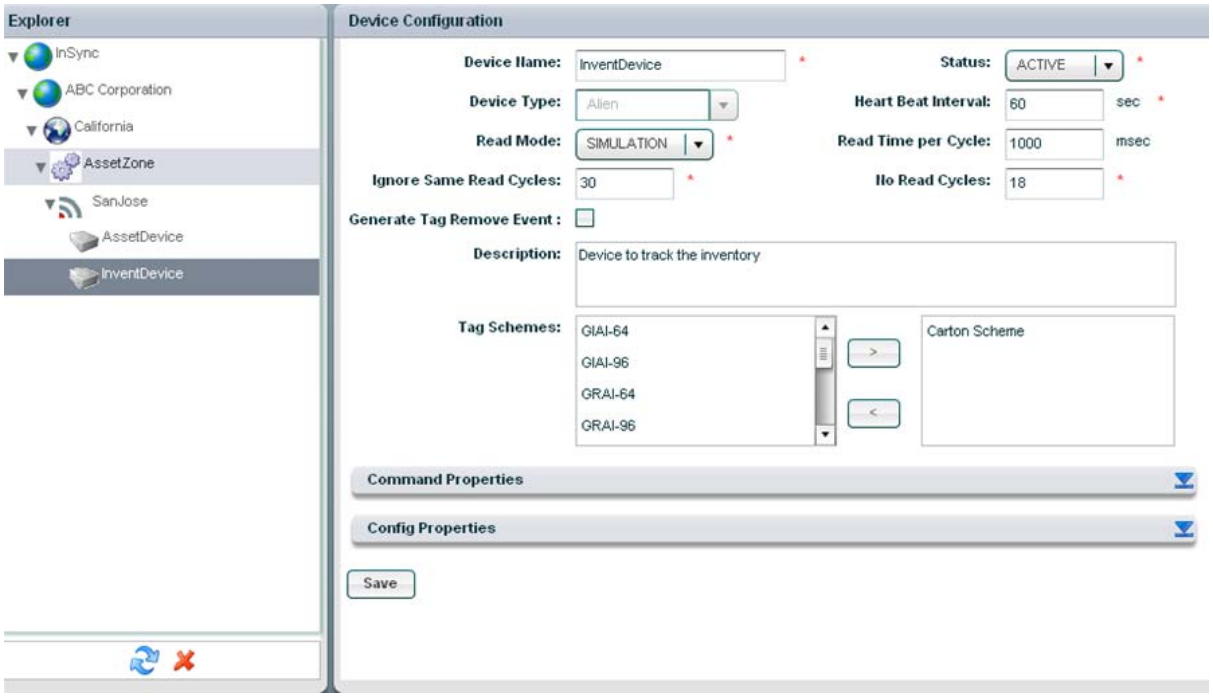
Option 2: (Add Device)

- Check the radio button against 'Create Device of Type'.
- Select the Device Type from the drop down.
- Click Create.



- Enter the details and click 'Save' at the end of the page.

➢ Device with the name is added to the left side tree.



Note: The difference between using option 1 and option 2 while adding a device is that in Option 1 the device is already created while in Option 2 we create the device with the name. The rest of the attribute configuration remains the same for both the options as described in Option 1.

Operations possible on a Device are

- Reload Device
- Delete Device

Note: The Device should be reloaded each and every time the properties change for the changes to get affected.

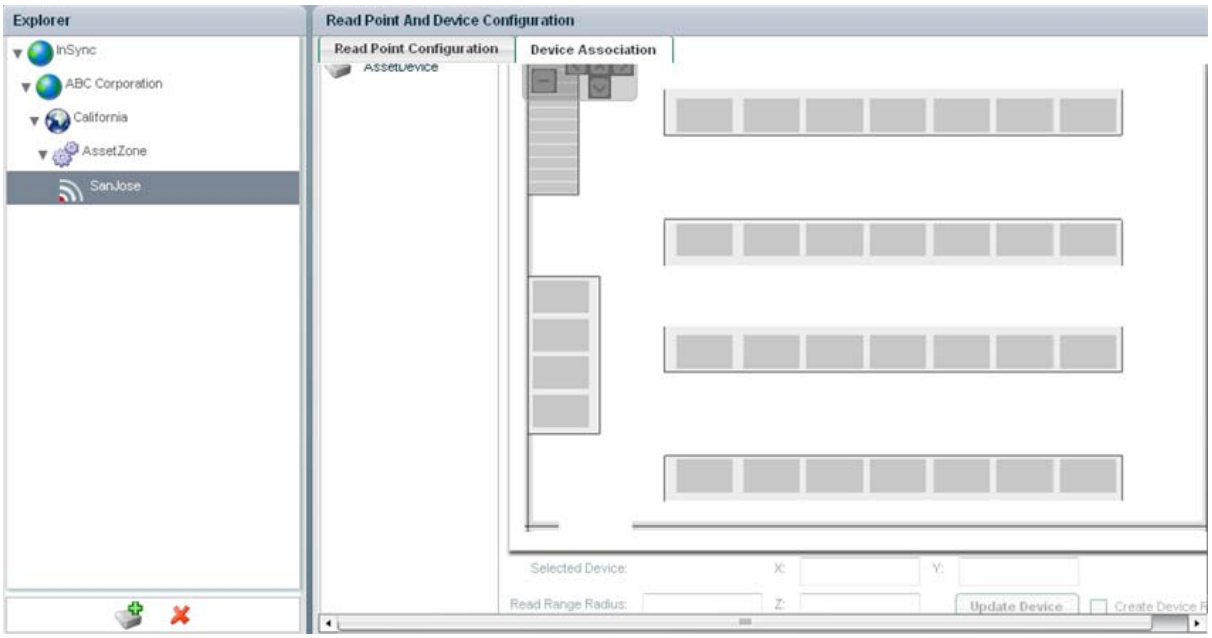
4.4.6 Device Association

The device association tab will be enabled if an image exists for the location and is calibrated. If the image exists and is not calibrated then the tab is disabled.

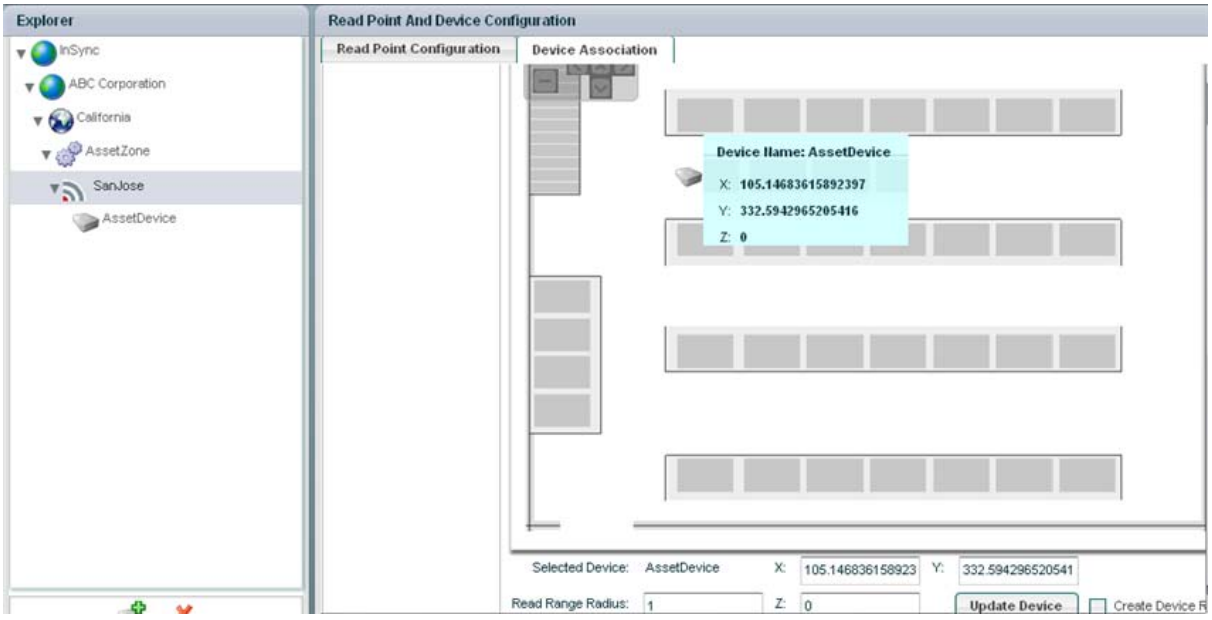
The purpose of the device association tab is to be able to associate the devices (created if any) to the Read Point by placing the device at the required co-ordinates on the calibrated image.

The devices can be created by selecting the read point and clicking on Add Device (the device gets associated to the read point) or can be created from the Devices tab discussed in the later sections.

Click on the Device Association.



1. The Device Association page lists all the un-associated Devices in the Enterprise which can be dragged and dropped on the calibrated image at a desired location.



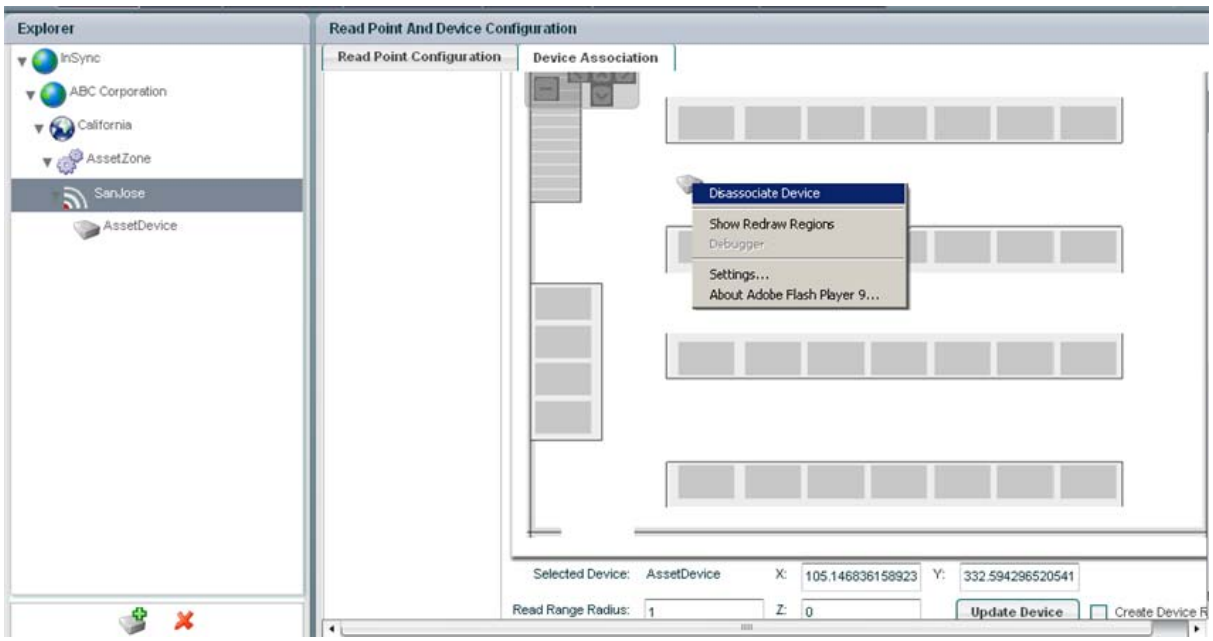
2. Select the device placed on the calibrated image. The following attributes are shown.

Attribute	Description
Selected Device	Name of the selected device
X	X Position of the device
Y	Y position of the device
Read Range Radius	The radius within which the device will be active.
Create Device Region	Selecting it will draw a region for the device on a RTLS viewer.

3. Click on Update Device to complete the association process.
4. Mouse over the device to view the device details in the tool tip.

To Dis-Associate the device

5. Right click on the Device to be dis-associated
6. Click on Disassociate Device



7. On Dis-Associating the device, it will be available for association again within the enterprise.

4.4.7 Add Edge Process

Edge Process hub forms the basis for all sensor-driven processes that allow collaboration across various locations. Once setup, the Edge Process hub acts as the filter to transform the raw data into events that are relevant to any business process. Acting as a tool for all parties in a business system, the Edge Process hub allows all groups to monitor, take inventory, and optimize business logic.

An Edge Process is a means by which a physical process can be described in terms of that states, transitions and actions.

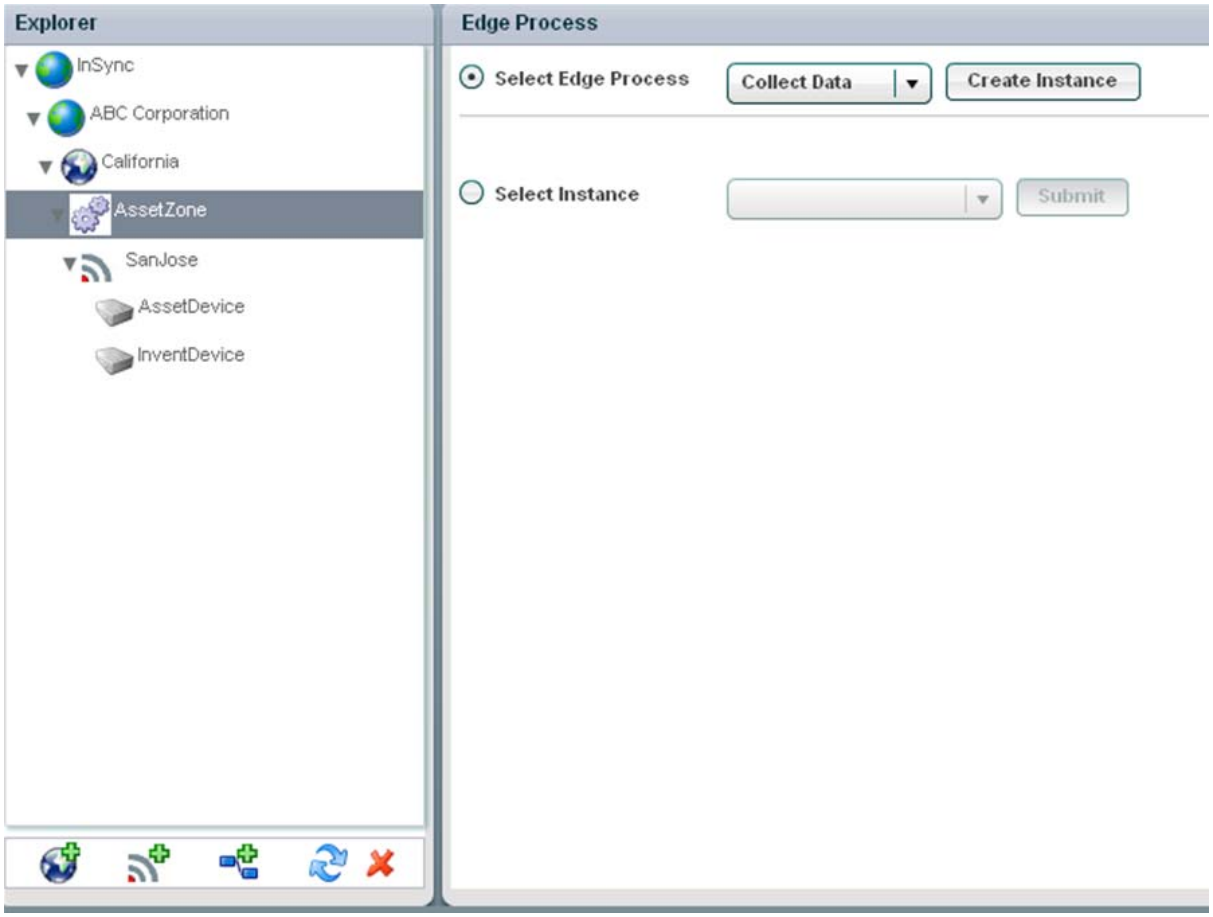
- A state stores information about the past, i.e. it reflects the input changes from the start to the current date.
- A transition is a state change and is described by a condition that needs to be fulfilled.
- An action is an activity that is to be performed.

Note: For a better understanding of Edge Process please refer to IApp Edge Process section.
In summary, following are the steps to configure an Edge Process using the IApp™ configuration.

1. Describe the process in terms of States, Actions and Events
2. Choose one of the provided Action Groups to perform the desired Action(s) in each State.

3. Based on the outcome of the Action(s), generate an event which causes the process to transition from the current state to another state.
4. Post appropriate events at key points in the execution of the process

Select the Process Zone and click on button to add an Edge Process.



There are 2 options that are possible while adding an Edge Process to a Process Zone

1. Option 1- Select Edge Process from the list of Master Edge Processes and create an instance of it.
2. Option 2- Select and use an Edge Process instance already created and which are not associated to Process Zone

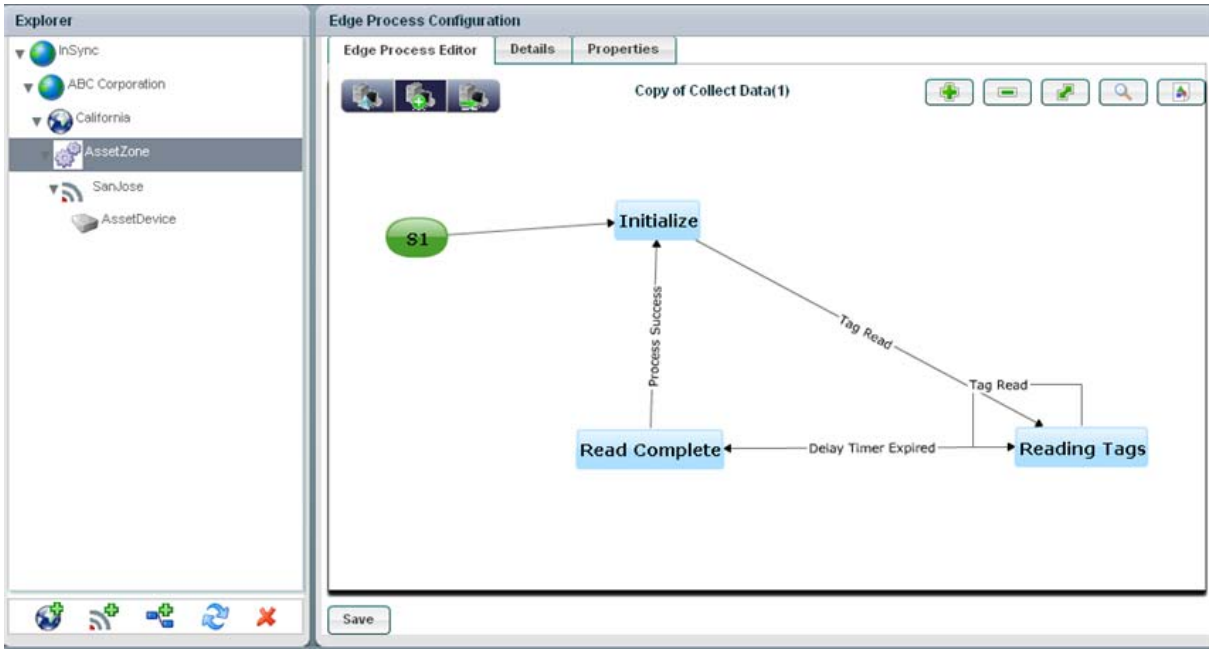
IApp provides 2 System Edge Processes by default in the system that can be used.

- Collect Data Process
- Pack Process

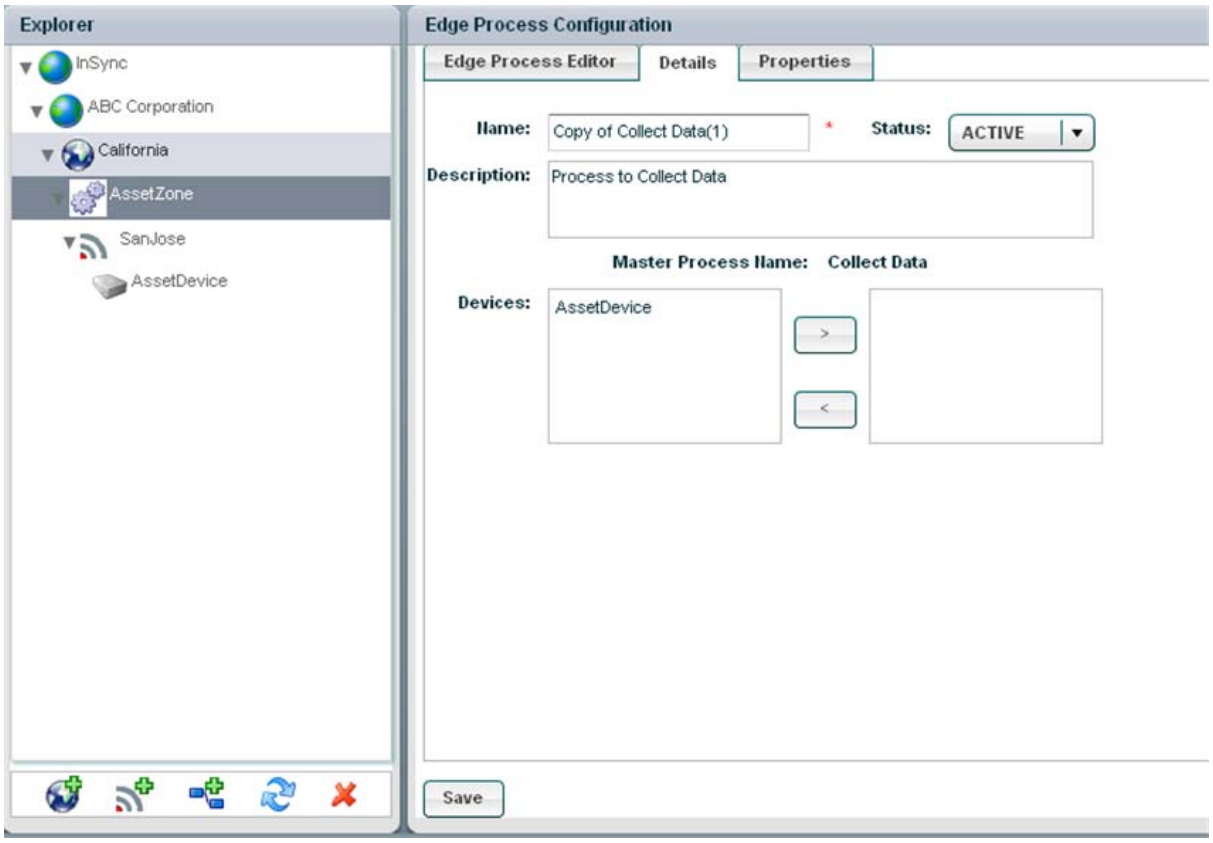
Refer to Edge Process section for information on Master Edge Processes and Instances.

Option 1:

- Check the radio button against ‘Select Edge Process’
- Select the Master Edge Process from the drop down that you want to create an instance for.
- Click ‘Create Instance’

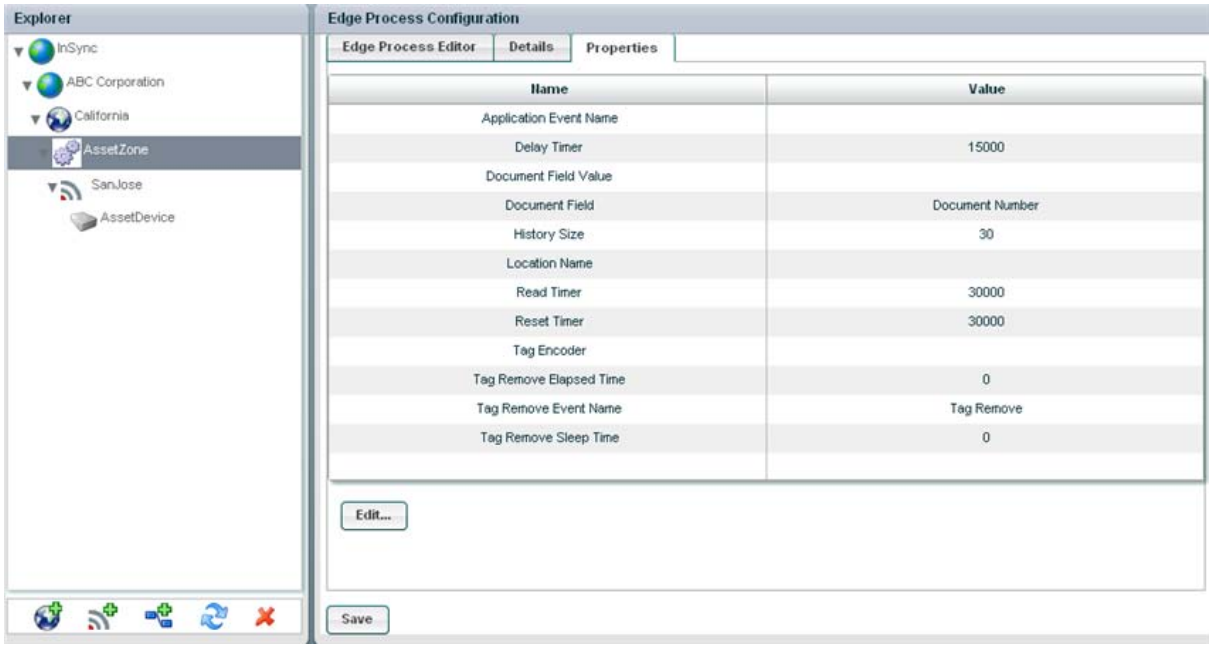


The **Details tab** displays the following attributes:



Attribute	Description
Name	Name of the Edge Process. Change the default name. The Edge Process name is unique across all Enterprise Organizations meaning you cannot have the same name in the same Enterprise or other Enterprise organization
Status	Denotes whether the Edge Process is Active or Inactive.
Description	Short notes
Devices	Map the devices configured to the Edge Process. The left side list box displays all the Devices configured. Use the > and < button to select or deselect the devices from the list.

The **Properties Tab** displays the following

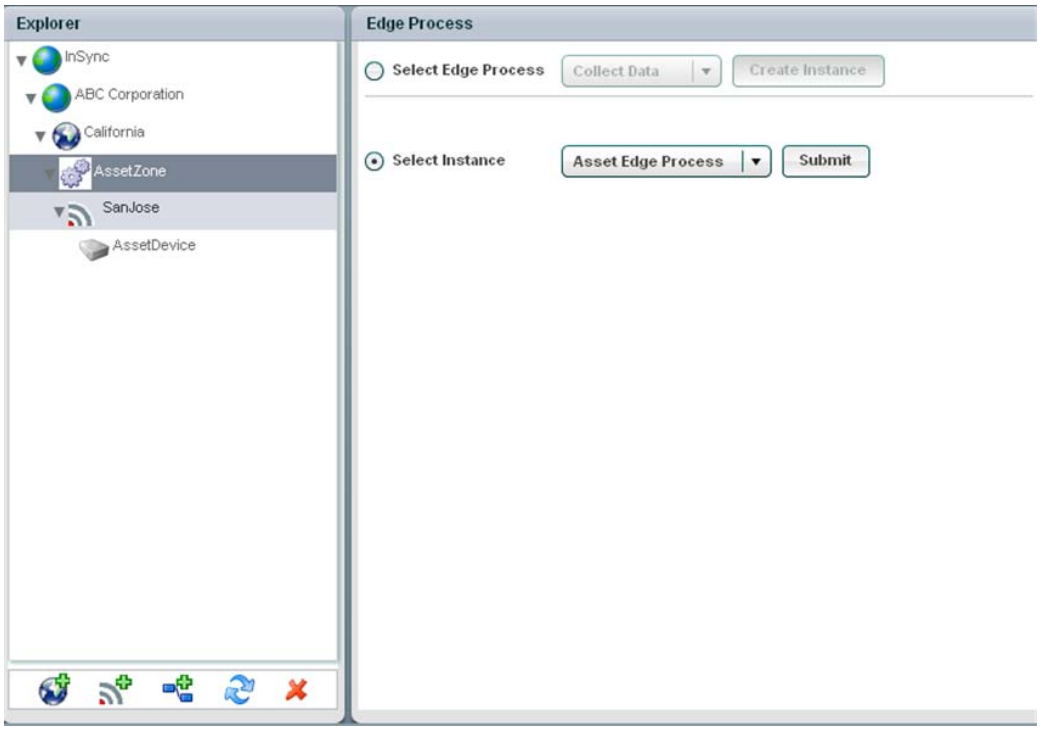


The Properties act as the input to the Transition Actions. The Properties for the Edge Process are defined in the Control Object and if used in the Actions then the value for the property is set here. In the absence of the value the default value set by the Control Object is used by the Action.

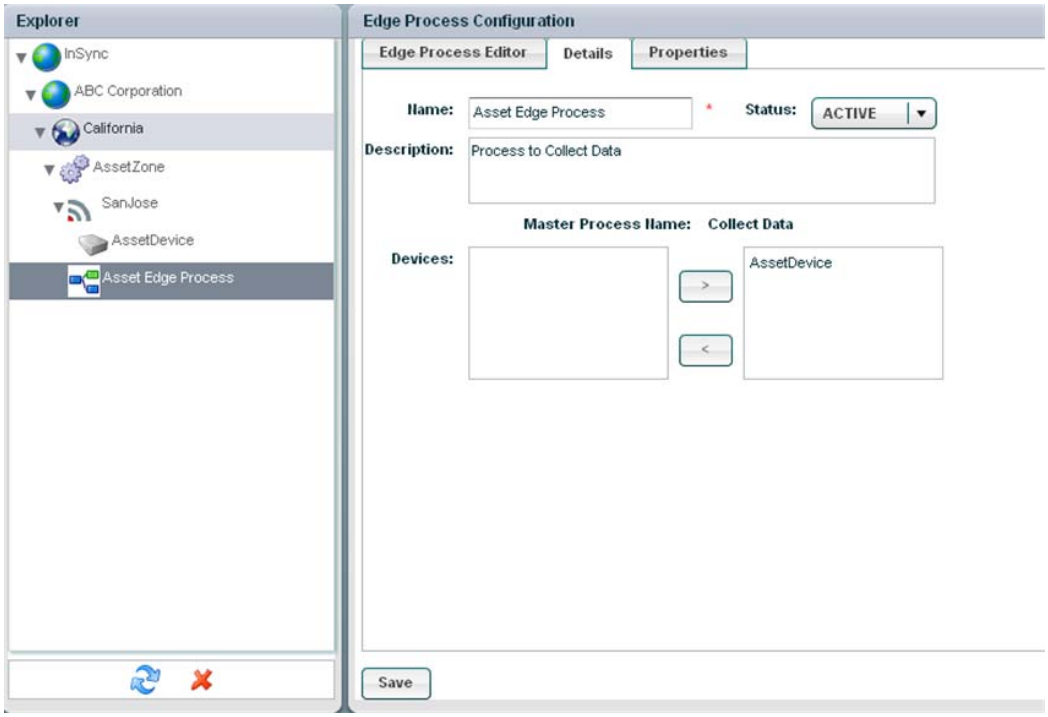
Note: To create a new Edge Process refer to Edge Processes Section

Option 2:

- Check the radio button against ‘Select Instance’
- Select the Edge Process Instance from the drop down.
- Click ‘Submit’



- The Edge Process is associated to the Zone.



- Refer to Option 1 for all the attribute definitions.

Note:

1. The difference between using option 1 and option 2 while Associating an Edge Process to a Process Zone is that in Option 1 an instance is created out of a Master Edge Process while in Option 2 an instance of the Edge Process already created is associated to the Process Zone.

2. Master and Instance Edge Processes are managed from the Edge Process Tab which will be discussed in the later sections.

Operations possible on Edge Process are

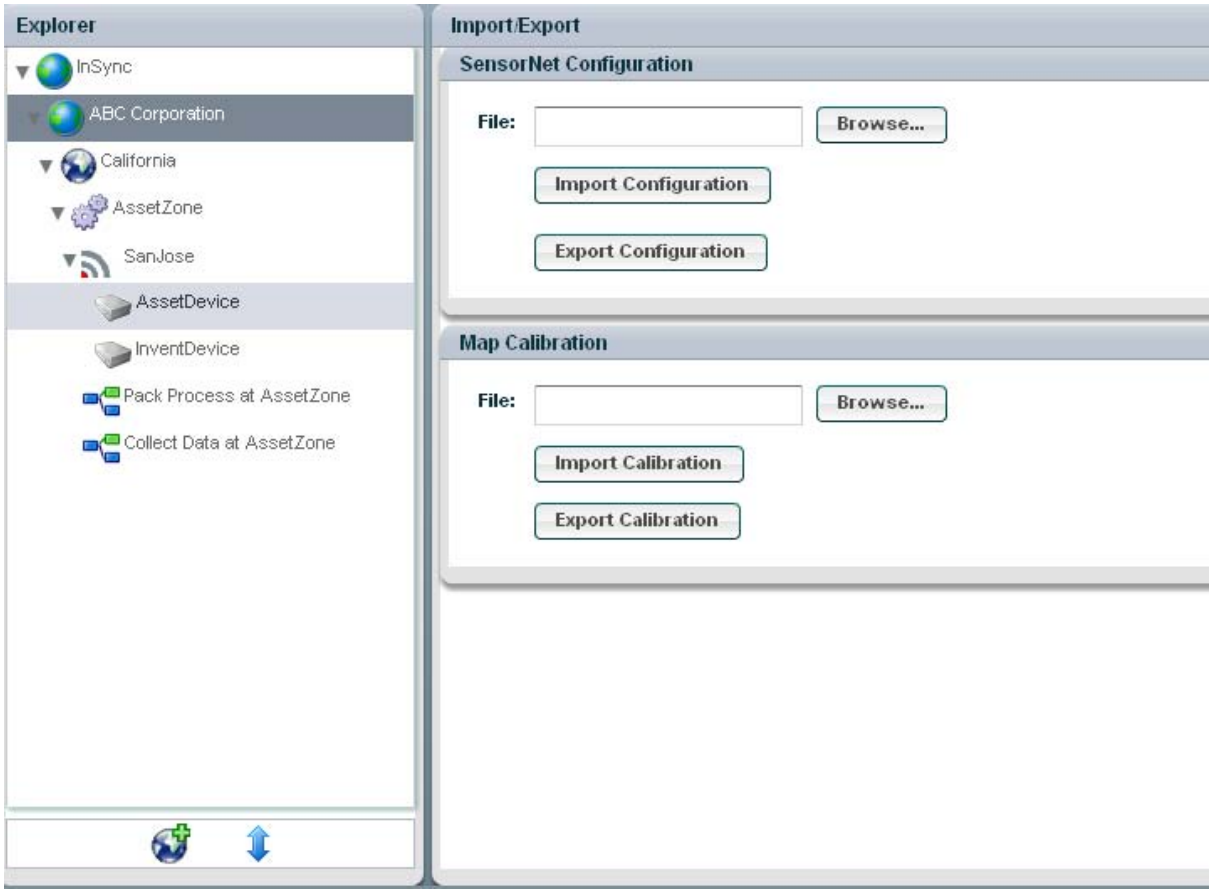
- Reload Edge Process
- Delete Edge Process

Note:

1. The Edge Process should be reloaded each time any of the attributes change for the changes to get affected.
2. The Delete Operation on the Edge Process does not physically delete the Edge Process but only disassociates itself from the Process Zone. Such Edge Processes will be available to be associated with the Process Zone.

4.4.8 Configuration / Map Data Import Export

The entire Configuration of an Enterprise Organization can be exported or imported by selecting the Enterprise Organization in the Overview left side panel and clicking on button.



- To Import the configuration Browse to the file to be imported and click ‘ Import Configuration’
- To export the configuration click ‘Export Configuration’ and navigate to the folder where you want to save the file. A file with the name OrgName_Configuration.xml gets created and stored at the desired location.
- Export saves the following entity information to the file
    - Locations
    - Read Points
    - Device Types
    - Process Zones
    - Device Types
    - Tag Schemes
    - Edge Processes

Follow the same steps to do an Export / Import of Map Calibration data.

Pre-Requisites:

Prior to importing the data it is required that the following information is already available in IApp.

- Enterprise Organization structure
- Catalogs

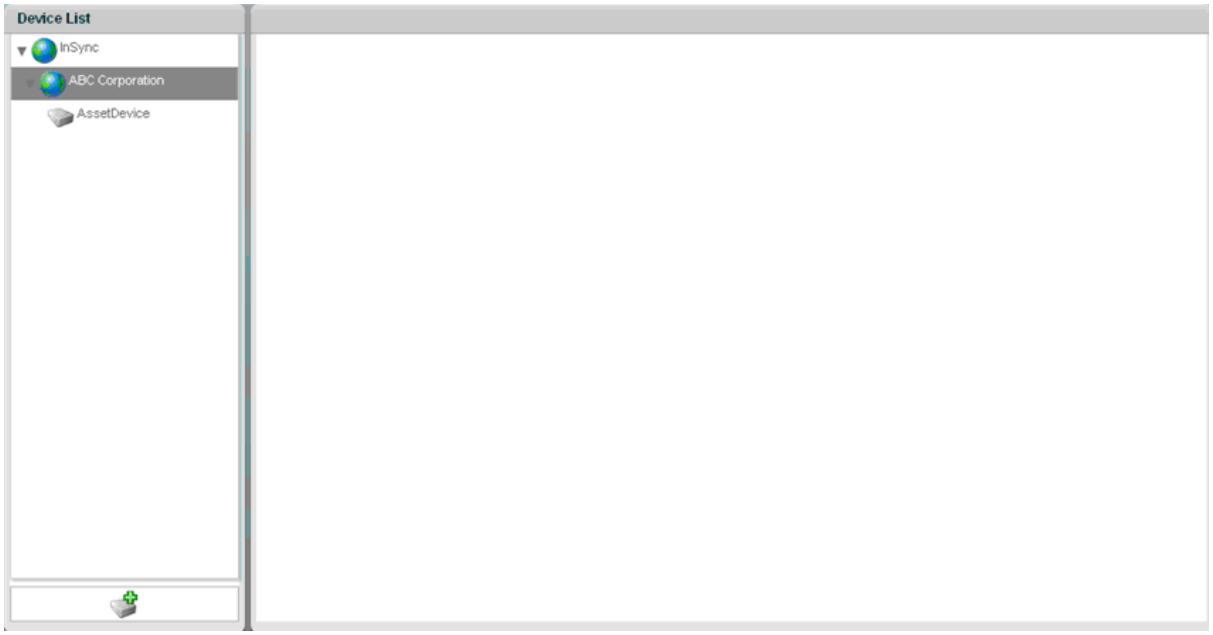
If the above information does not exist then it is recommended to export and import the above mentioned data in the same sequence as listed above.

Note: For more information on Export / Import please refer to the section on Export / Import.

4.5 Devices

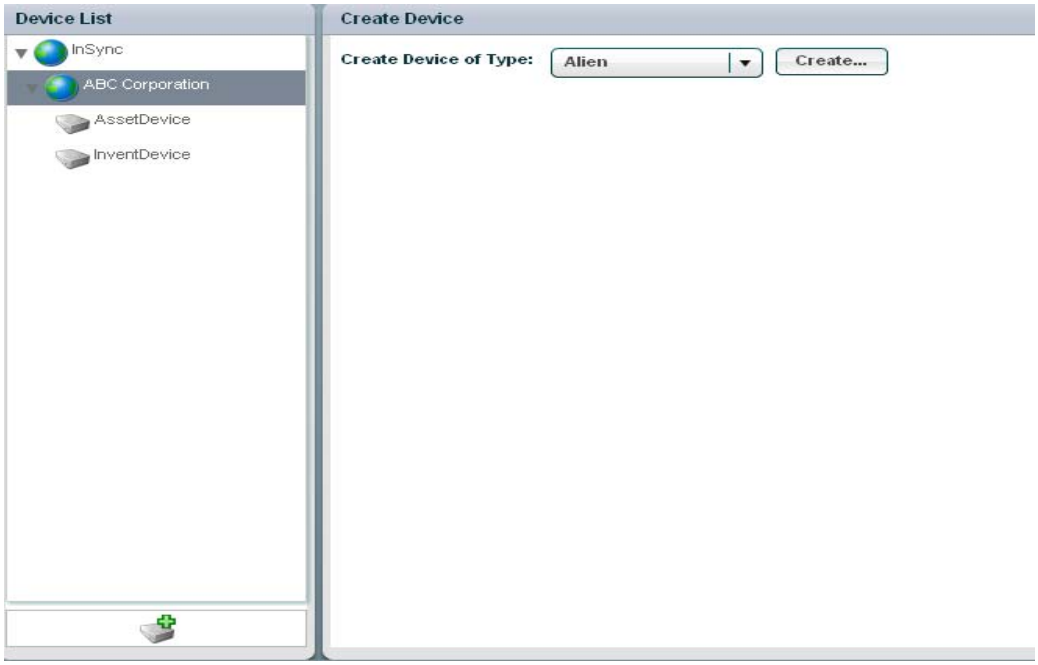
Device stores all the reader properties, communication protocol and the mode of operation. The device configuration is responsible to initialize the data at the beginning of the edge process, monitor the connectivity with the physical reader and provide interface to other modules in the system to access the device. It also provides the first level of tag validation to ensure the data is proper.

To add Devices click on ‘Devices’ menu item in the floating menu.



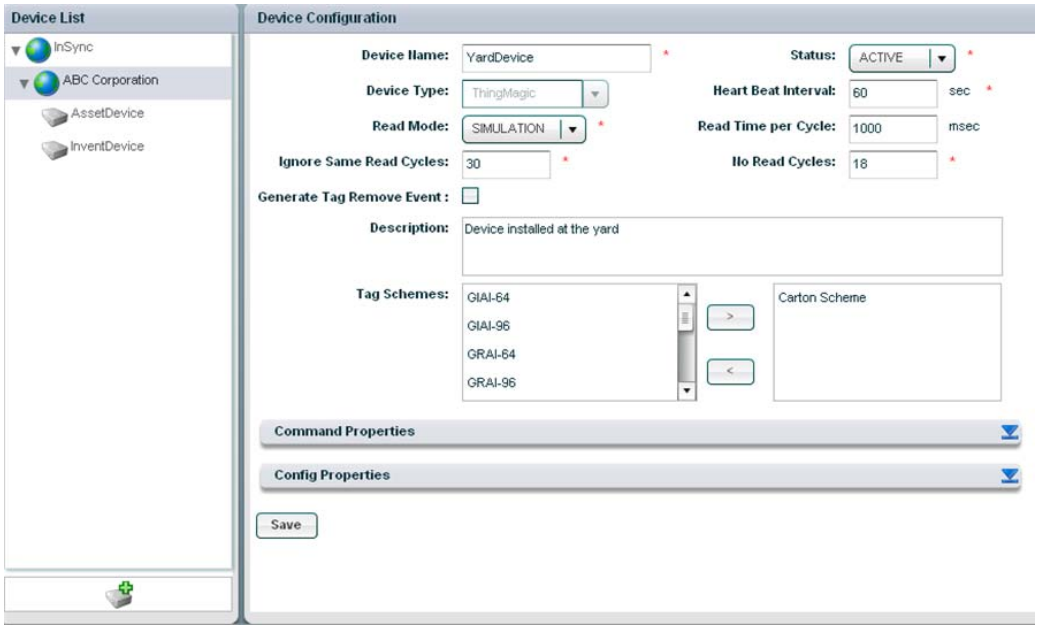
The Devices panel gives a listing of all the devices existing within each Enterprise Organization. Logged in Users belonging to the Enterprise organization will be able to add new devices and modify the existing device details in that Enterprise only.

To add a Device, select the Device in the left side panel and click on in the buttons panel.

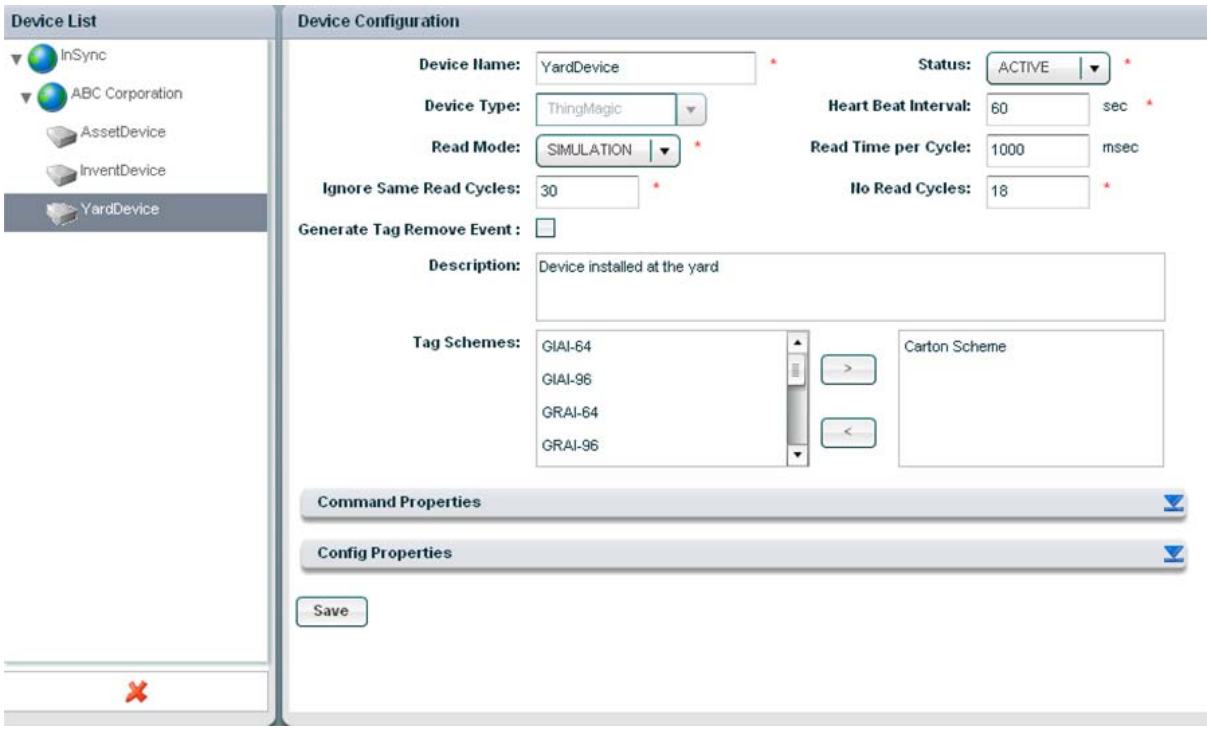


- Select the Device Type from the drop down.
- Click Create.

Note: Adding new Device Types is an administrative task and is accomplished from the Software tab accessible only to the Home Enterprise administrative user.



- Provide the device attribute information.
- Click Save. The Device gets added to the tree in the left side panel.



The attributes for the Device are as follows.

Attribute	Description
Name	Name of the device. The Device name is unique across all enterprises. Device with the same name cannot exist in more than one enterprise organization.
Status	Denotes whether the Device is Active or Inactive
Device Type	Type of Device being configured
Heart Beat Interval	Interval at which the system monitors the health of the device
Read Mode	Mode to which the device is tuned. • Auto mode polls the Reader for reads continuously until the Reader is turned off or the application is shutdown • Semi-Auto needs the command to be sent in Manual mode at least once before it goes into Auto mode • Manual mode needs the command to be explicitly sent to the Reader; this can be done in Activities tab discussed later. • Simulation mode imitates the Reader's execution of commands; this is useful in development and test environments. **Note:** For Gen2 commands to work, the device has to be in **Manual** Mode ONLY.
Read Time per Cycle	It is the period in milliseconds in which the reads are read. For ex; if set to 2000ms then the read cycle is 2 seconds meaning the reads are read every 2 seconds
Ignore Same Read Cycles	This is the number of cycles within which the read will be ignored/filtered if it appears again within the cycle from the first cycle it is read. For ex; if set to 10, then the reader will ignore the read if it appears more than once in the 10 cycles starting from the first cycle it is read. Beyond the 10 cycles it will be considered as a new read.
No Read Cycles	If a tag is read in a cycle and no reads happen for the cycles mentioned from the first cycle it was read then the reader considers it to have gone out of its sight and generates a Tag remove event.
Generate Tag Remove Event	If checked then the event is raised based on the 'No Read Cycle' time configured else ignored.
Tag Scheme	Defines the encoding scheme of tag value fields. Select the desired tag scheme from the list box.

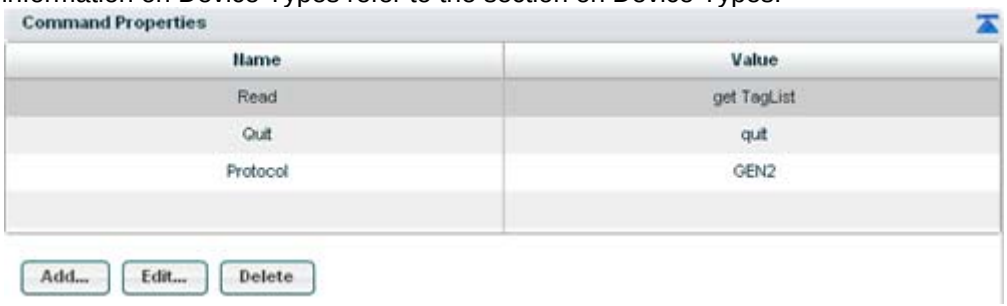
	For more information on tag Schemes refer to the section on Tag Schemes.

Command Properties:

These are device commands along with the values in the form of name value pairs. The names could be variables that are defined in the Device Type inside [] brackets.

Use the min max button in the Command Properties panel to view the command properties. The panel provides means of adding, editing or deleting a command property.

For more information on Device Types refer to the section on Device Types.



Protocol is specified as 'GEN2' by default. You can change the value if you want to execute commands in a Protocol suitable for your device. Most Readers support multiple protocols; valid protocols are ISO18000-6B, EPC0, EPC1, and GEN2.

Config Properties:

Config properties are used to configure and fine tune the device to run properly. Property names are already populated based on the Device Type.

Use the min max button in the Config Properties panel to view the Config properties. The panel provides means of adding, editing or deleting a command property.

Common Properties of the device- IP address, Antenna count, Send Read Event, Use Device Time, Port Number are configured here.

For more information on the Config Properties for each Device Type refer to IApp Device Types section.

Operations possible on the Device are

➢ Delete

1. Select the Device to be deleted and click to delete the device.
2. Deletion of the device is not possible if the device is mapped to an Edge Process.
3. To delete such Device, disassociate the device from the Edge process first before continuing with the operation.

4.6 Tag Schemes

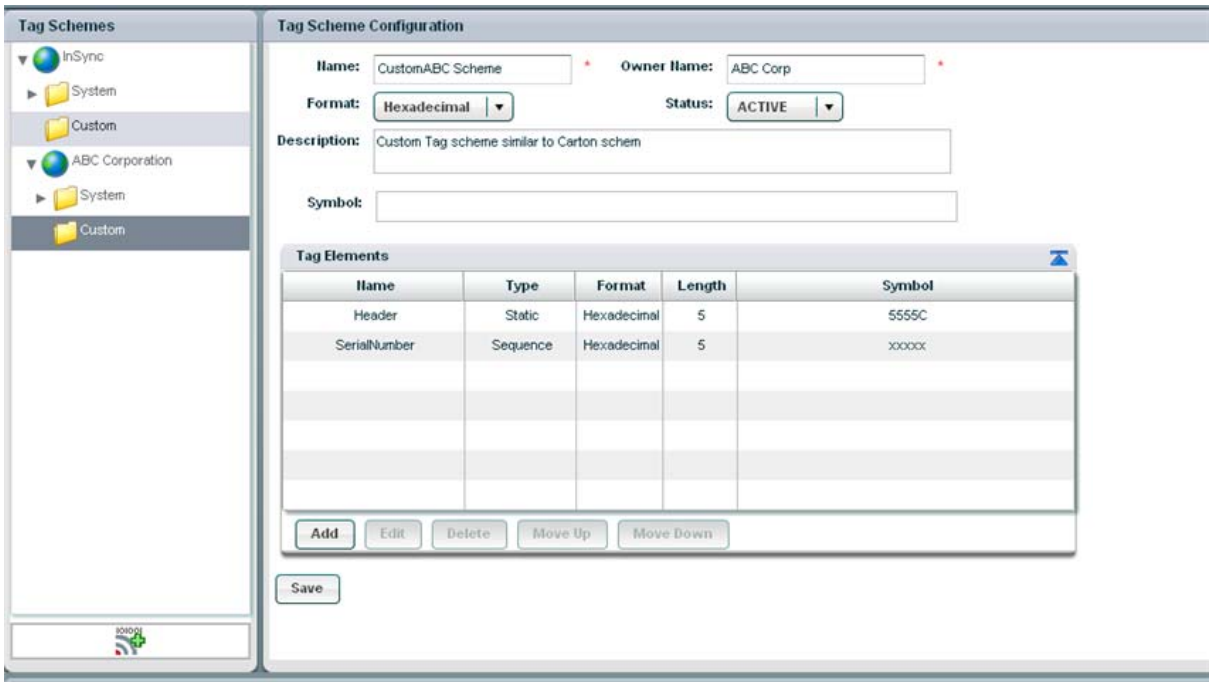
The Tag Schemes define the encoding scheme of tag value fields. Encoding scheme indicates name and format for each field in the tag data. Tag data can be a combination of Product Id (7 characters) and Serial Number (8 characters) each separated by a delimiter like (.)

The Tag Schemes in IApp are listed by Enterprise – System / Custom.

- The System tag schemes are provided by default and are available across all Enterprises.
- Only users with Administrative rights at the Home organization will be able to add or modify the System tag schemes
- Users belonging to an Enterprise organization will be able to add new Custom tag schemes and modify the existing custom tag schemes in that Enterprise only.
- Administration users at Home Organization will be able to add Custom tag schemes in other Enterprise organizations.

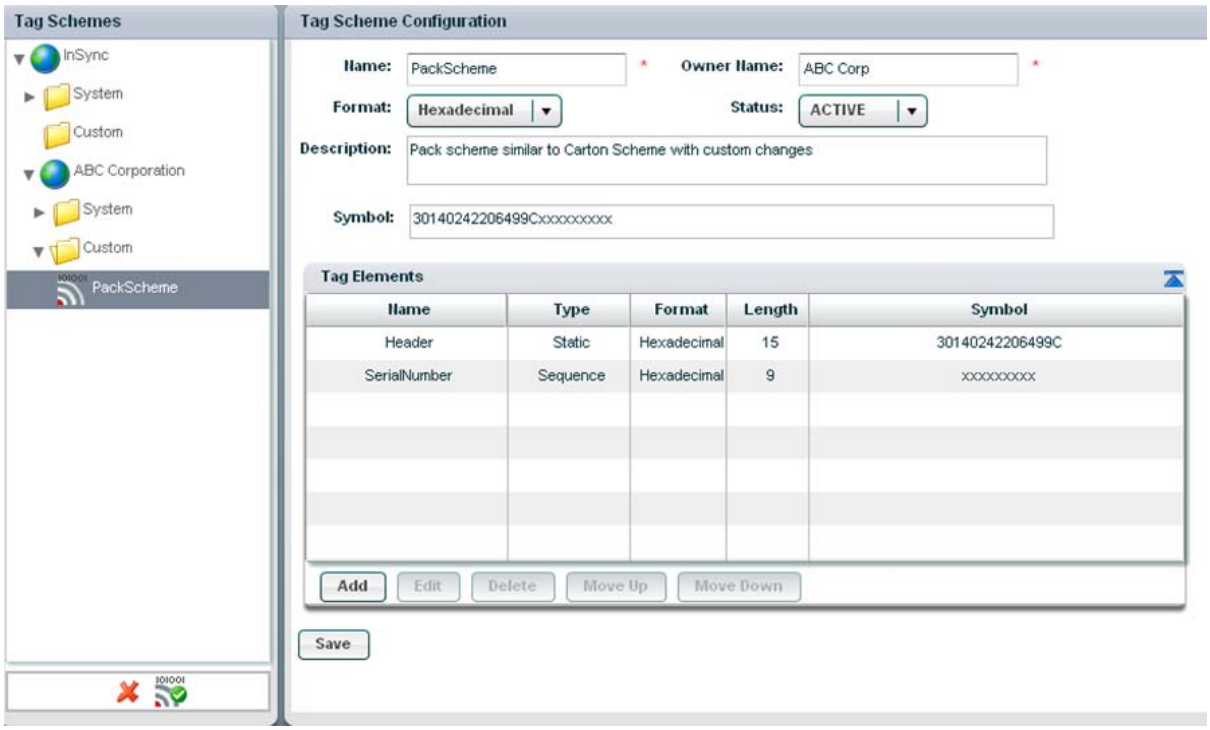
To create a tag scheme

1. Select the Organization
2. Select System or Custom based on the access.
3. Click



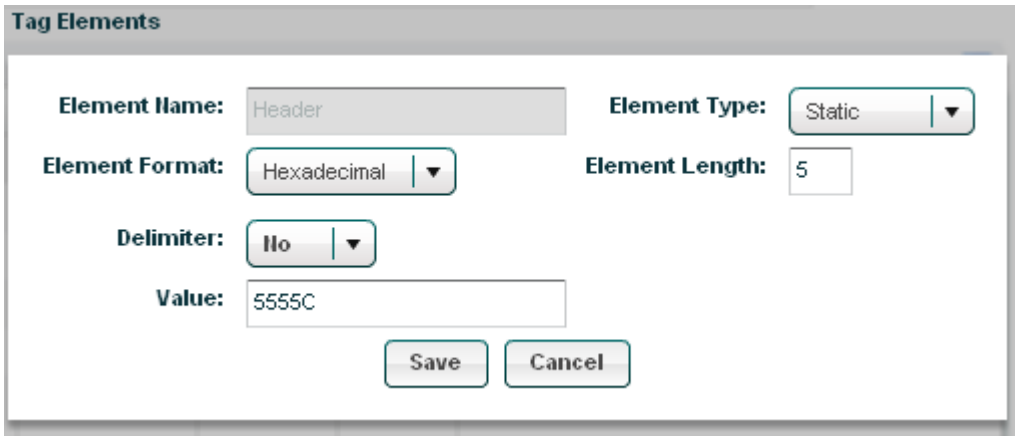
Attributes of the tag scheme are:

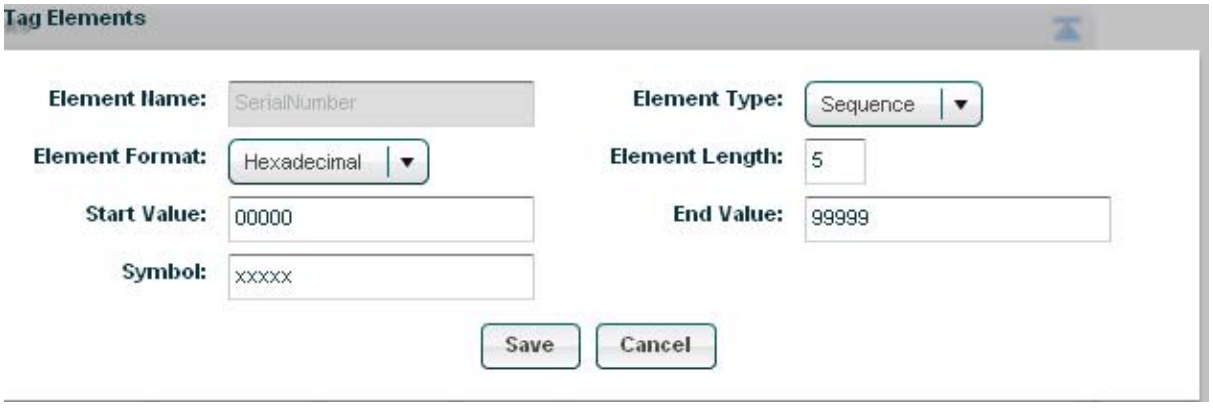
Attribute	Description
Name	Name of the Tag scheme. The name is unique within an Enterprise organization
Owner Name	Owner who owns the tag scheme.
Format	Data format to define the tag scheme. It could be • Hexadecimal – Characters in Hex format (0-9), (A-F) • ASCII – representation in numbers • Binary – Compressed data • Digits – Numerical data
Status	Denotes whether the Tag Scheme is Active or Inactive
Description	Short Description for the tag scheme
Symbol	Final Format displaying the element structure after configuring the tag elements. Note: The Symbol gets automatically added once a Tag Scheme is saved. The Tag Elements Symbols are appended and shown.
Tag Elements	The Tag elements define the various structural elements of the tag scheme.



Tag Elements

The Tag elements define the various structural elements of the tag scheme. To add the Tag elements while creating the tag scheme click on Add button.





The following table lists the attributes of the Tag Elements

Attribute	Description
Element Name	Element name to be added
Element Type	**The type can be Static/Sequence** • Static – Constant element whose value does not change • Sequence – Used to define elements whose value increments sequentially
Element Format	**Format to define the Element. It could be** • Hexadecimal – Characters in Hex format (0-9), (A-F) • ASCII – representation in numbers • Binary – Compressed data • Digits – Numerical data
Element Length	**The Length of the Tag element being defined**
Value / Start Value / End Value	**Values based on the Element Type.** • For Static Type, capture the Value which is a constant and is of the element length defined • For Sequence Type, capture the Start value and End Value
Value / Symbol	**The Element Type decides the Value or the Symbol** • If the Type is Static then the Value will be the constant value • If the Type is Sequence then the Symbol will be the value based on the Start and End Value and will be of the Length defined.

- The combination of the Tag Elements (Static and Sequence) that are created will define the Symbol and the tag scheme in turn to be used
- The Tag Scheme is associated to the Device while creating a Device

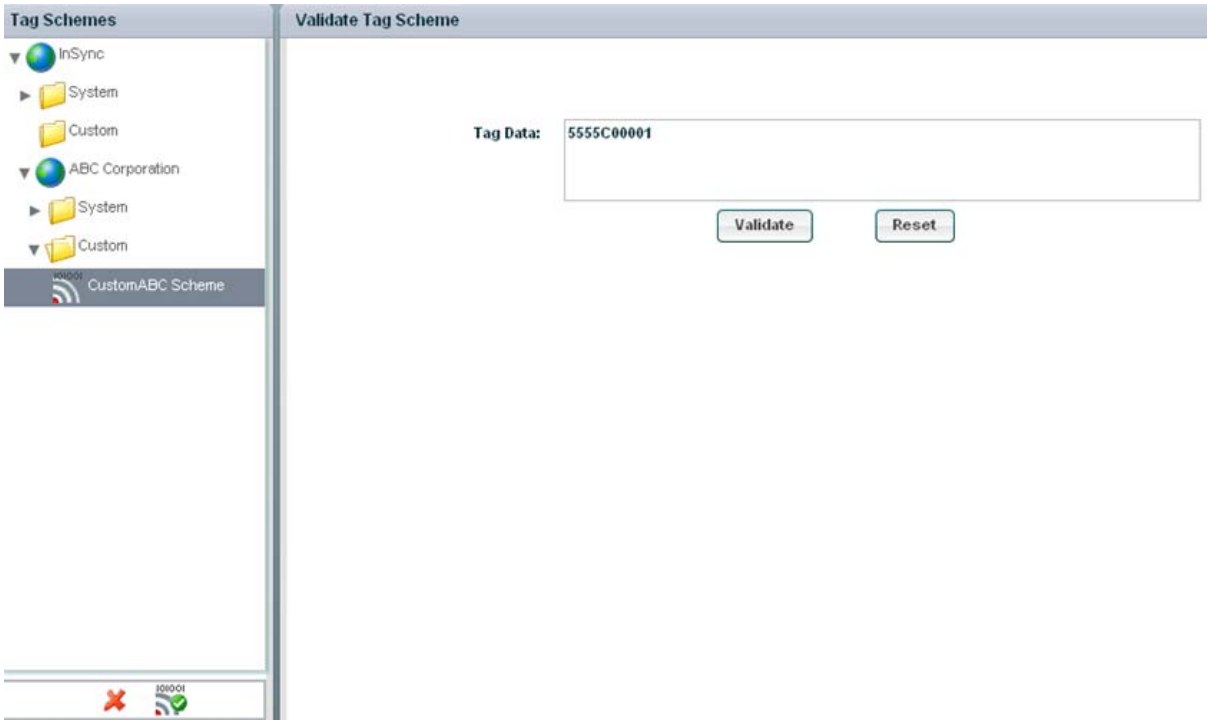
Operations possible on a Tag scheme are

➢ Delete Tag Scheme

Tag Schemes which are defined and not associated to a device will be allowed to be deleted. The tag Scheme should be disassociated from the device before proceeding with the delete.

➢ Validate Tag Scheme

The validate tag scheme is a verification tool to check the validity of the tag scheme that is defined based on which corrections could be made before it is configured to be used.



Eg. From the definition of the tag scheme i.e CustomABCScheme, the static element value is 5555C and the Sequence start and end values are 00000 / 99999 respectively. This would mean that the valid tag data reported by the device using this particular tag scheme would be 5555C00001 and sequence there on.

4.7 Edge Processes

An Edge Process is a means by which a physical process can be described in terms of that states, transitions and actions.

- A state stores information about the past, i.e. it reflects the input changes from the start to the current date.
- A transition is a state change and is described by a condition that needs to be fulfilled.
- An action is an activity that is to be performed.

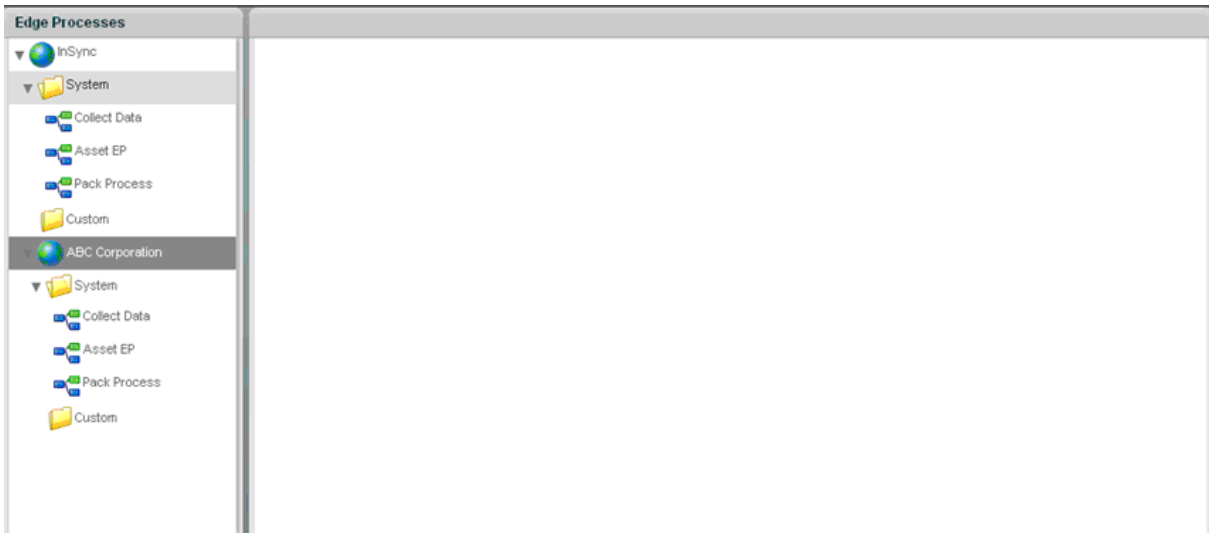
Note: For a better understanding of Edge Process please refer to Edge Process section.

In summary, following are the steps to configure an Edge Process using the Edge Process modeler.

1. Create a state.
2. On the created state configure the Action Groups provided to perform the desired Action(s) in each State using the General properties folder.
3. Set the pre-script and post-script based on the need.
4. Based on the outcome of the Action(s), generate an event which causes the process to transition from the current state to another state.
5. Post appropriate events at key points in the execution of the process

There are 2 ways of creating an edge process.

1. From the Explorer page described in section 4.1.5.
2. From the Edge Processes menu using the Edge Process Modeler.



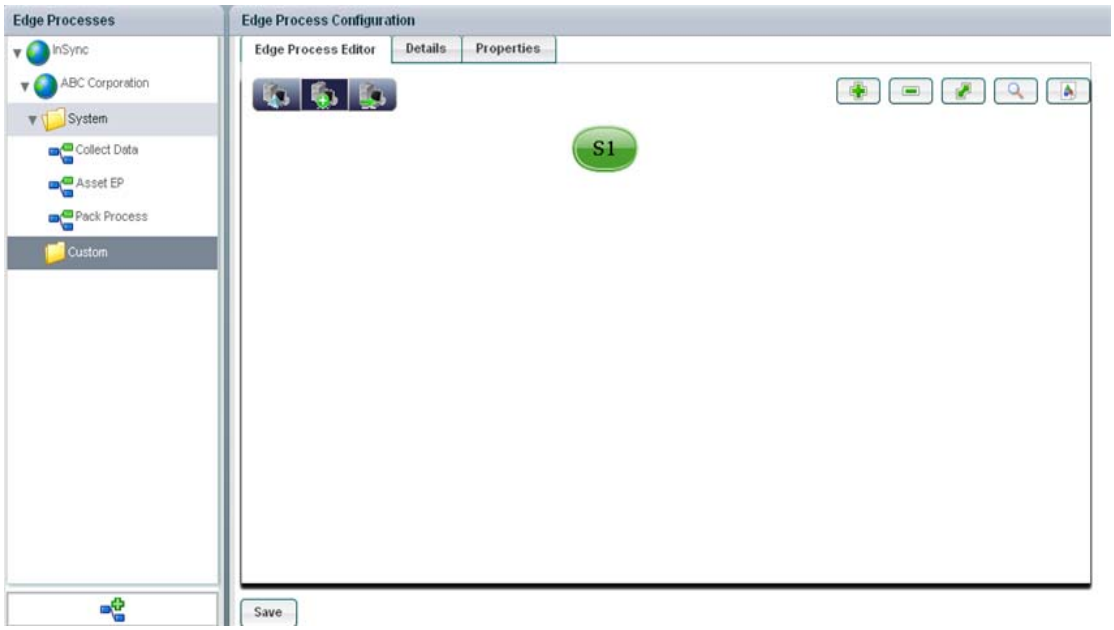
The Edge Processes in IApp are listed by Enterprise – System / Custom.

- The System Edge Processes are provided by default and are available across all Enterprises.
- The System / Custom Edge Process are all Master Edge Process from which instances can be created.
- Instances of the Edge Processes that are created in Section 4.1.5 are listed against the Master Edge Process that the Edge Process has been created from.
- All instances created can be managed by selecting the Edge Process. Managing an instance provides a means of updating from the Master if the master Edge Process has undergone changes or updates the Master itself with the changes made to the instance.
- Only users with Administrative rights at the Home organization will be able to add or modify the System Edge Processes.
- Users belonging to an Enterprise organization will be able to add new Custom Edge Processes and modify the ones in that Enterprise only.
- Administrative users at Home Organization will be able to add Custom Edge Processes in other Enterprise organizations.

12.4.1 Create Edge Process

To create an Edge Process

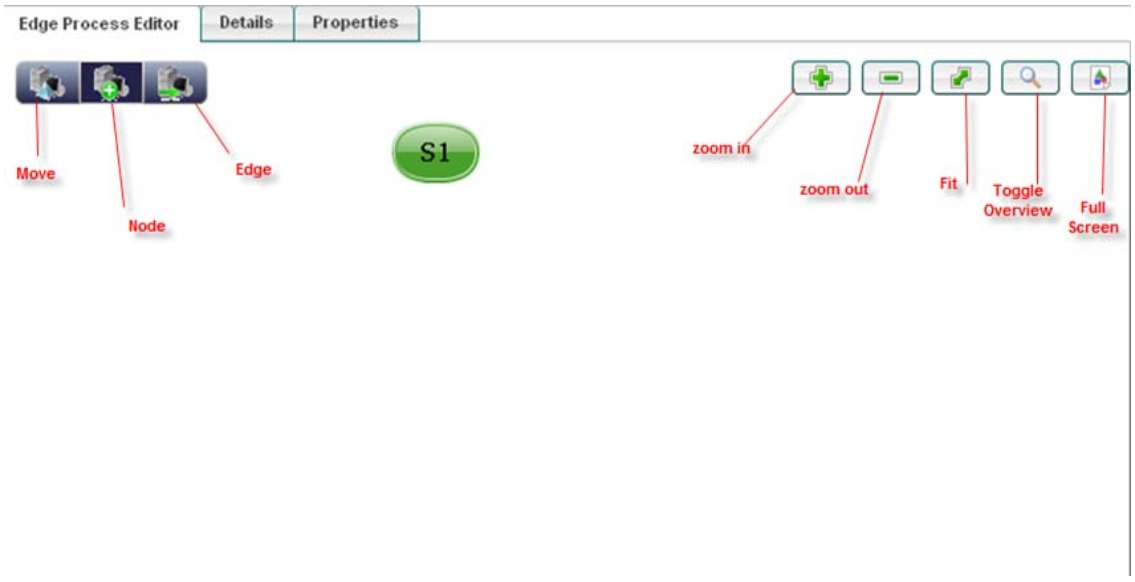
1. Select the Enterprise Organization
2. Select System or Custom based on the access.
3. Click to create an Edge Process



The modeler is split into 3 tabs.

- **A.** The editor panel shows the editor to create the states and transitions. An initial state S1 is created by default.
- **B.** The details panel contains the attributes (Name,Status,Instances) of the Edge process
- **C.** The properties panel captures the Edge process properties

Edge Process Editor

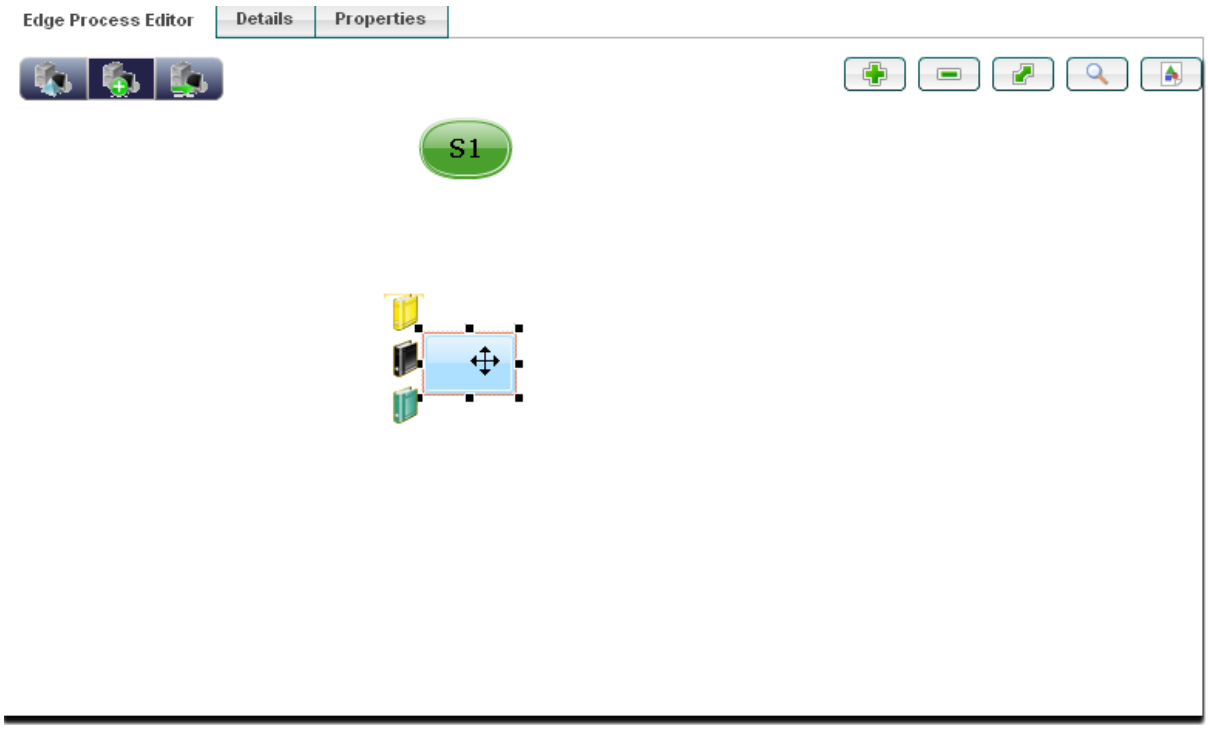


- Click on Move to be able to move the states on the panel.

- Click on Node to be able to add a new state as well as view the property folders on a created state

- Click on Edge to be able to create a transition between 2 states and set the actions on the transition.

To create a new state – select Node in the toolbar and click on the panel to create a state



- The state can be resized by selecting it and using the mouse to size it.
- The following properties can be set for each state.
- State Name/ Entry Actions using the General folder
    - Pre Script
    - Post Script and

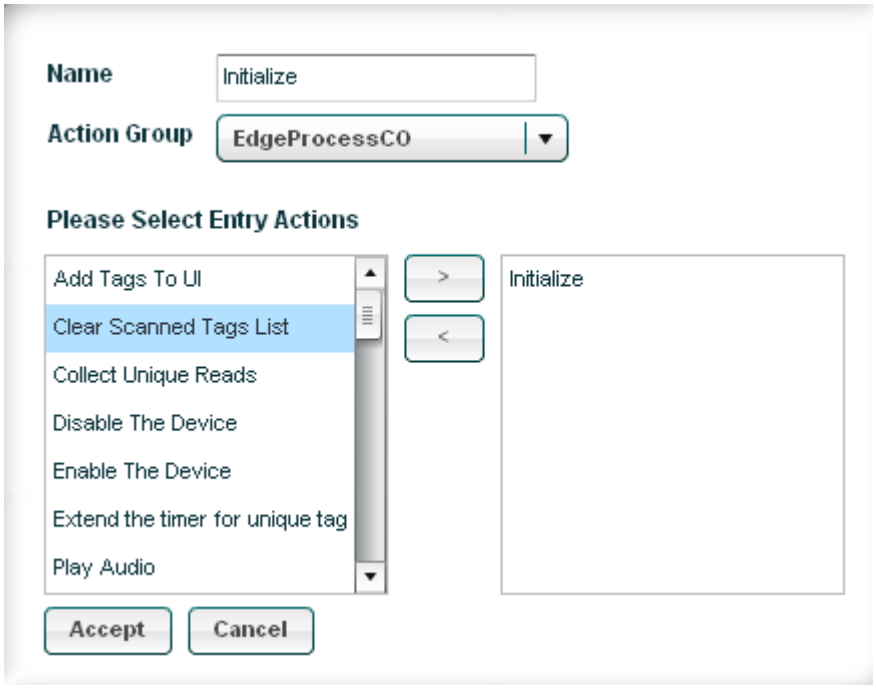
- Move the cursor to the state to see the 4 properties

➢ General Properties where State Name and Entry Actions can be set.

- Add Pre Script
- Add Post Script

Add Entry Action

- To create a state name and add Entry actions mouse over to the state on the editor panel. If the General folder is not visible click on before the mosue over.
- Select .

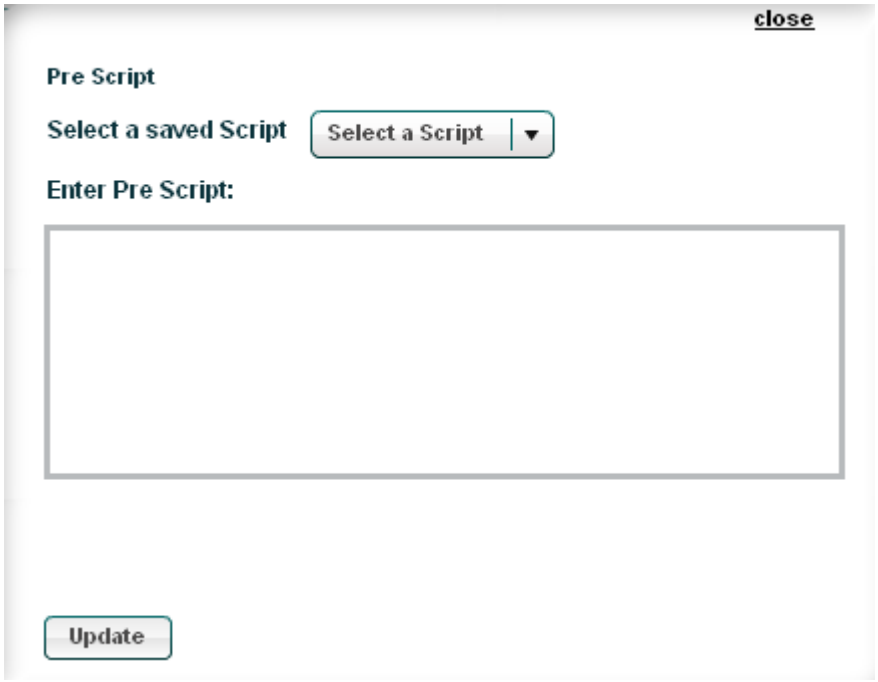


Attribute	Description
State Name	Name for the transition
Action Group	Group of pre-defined Controlled Objects each of which contains a list of actions that are available to be performed by a configured Edge Process.
Entry Action	Actions supported by the selected Control objects selected from the Action Group

- Click Accept to save the state and close the panel.
- Click cancel to cancel the operation.
- Follow the same steps to update an existing state.

Add Pre Script

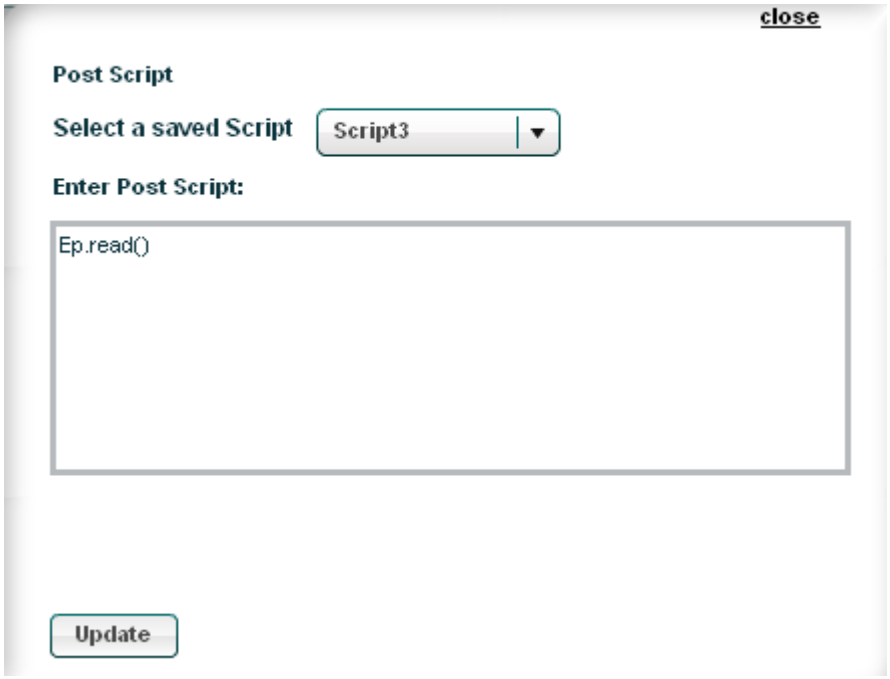
- To add a Pre Script, click on in the toolbar and mouse over to the state on the editor panel
- Select .



- Select a script from the pre defined scripts from the drop down or
- Enter the script in the Pre Script text box
- Click Update and close to exit the popup.
- Repeat the same steps to edit the script

Add Post Script

- To add a Post Script, click on in the toolbar mouse over to the state on the editor panel
- Select .



- Select a script from the pre defined scripts from the drop down or
- Enter the script in the Post Script text box
- Click Update and close to exit the popup.
- Repeat the same steps to edit the script

Create Transition

A transition can be created

- Between 2 different states or
- Within a single state itself.

Create Transition between 2 states

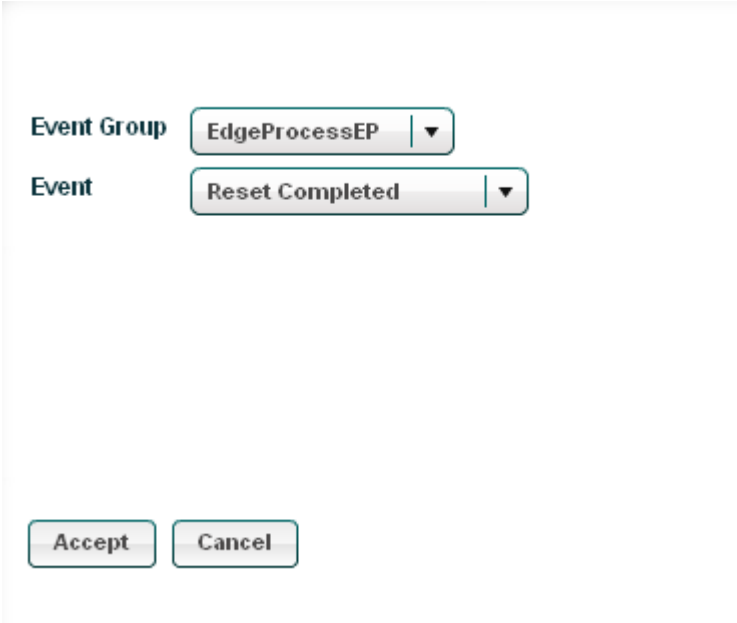
A transition is a state change and is described by a condition that needs to be fulfilled.

- Click on edge and mouse over to the state from where the transition will occur

- Drag the mouse to the 2nd state where the transition will end and release the mouse

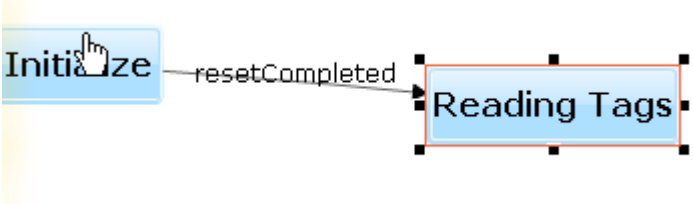
- As the mouse is released the transition is created between the 2 states.
- The event to trigger the transition can be added by mouse over to the transition line and click on

- Clicking on brings up the event panel to add the event



Attribute	Description
Event Group	It is a set of pre-defined Event names grouped together logically that can be used to effect an event transition from one state to another.
Event	Event name

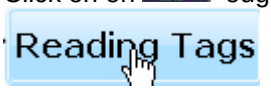
- Select the Event Group from the drop down
- Select the Event and click Accept.
- Click cancel to cancel the operation and close the pop up.
- The event action is seen at the transition



Create transition to the same state

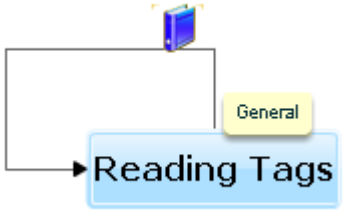
To create a transition for the state to itself

- Click on on edge and Mouse over to the state from where the transition will occur



- Drag the mouse out of the state and release it at the same state.

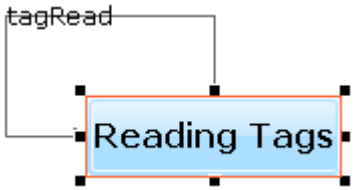
- As the mouse is released at the same state the transition is created.
- The event to trigger the transition can be added by mouse over to the transition line and click on

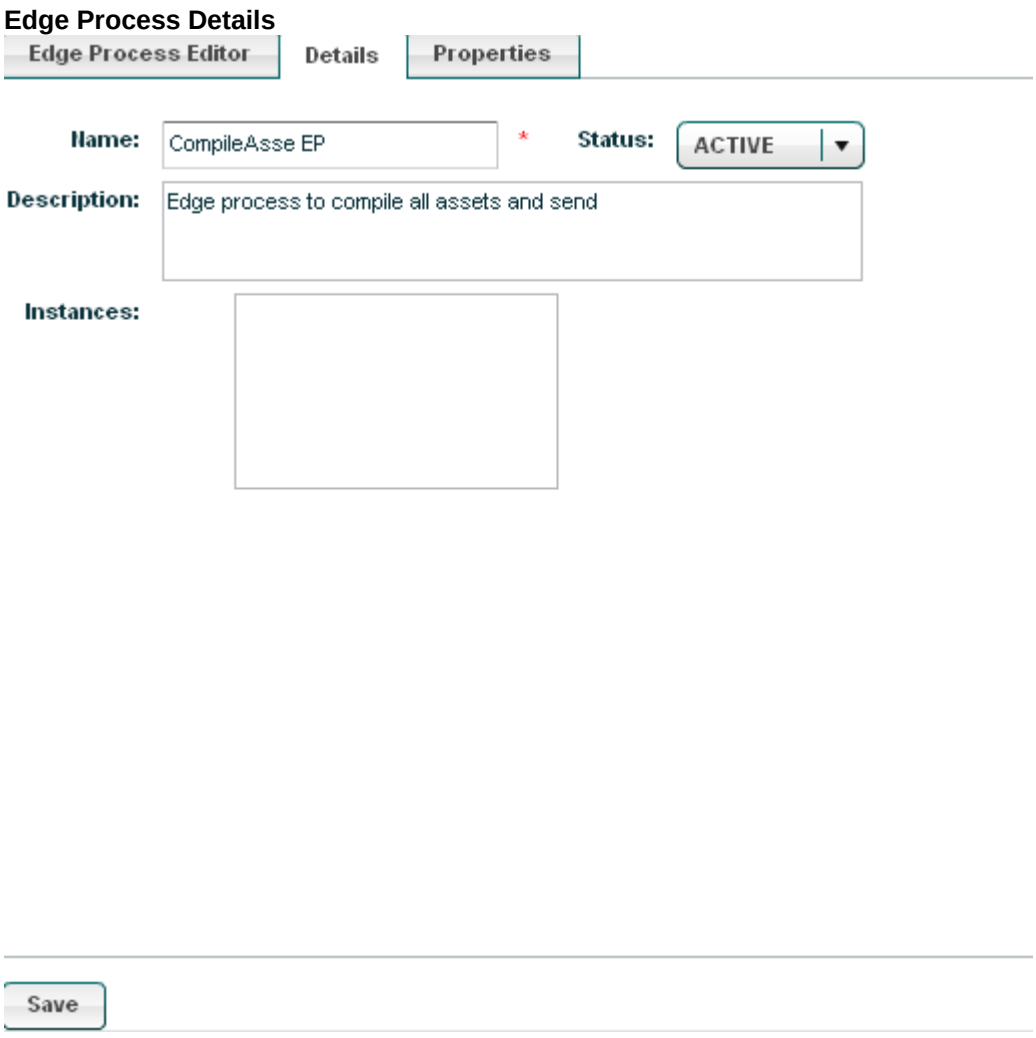


- Clicking on brings up the event panel to add the event

Attribute	Description
Event Group	It is a set of pre-defined Event names grouped together logically that can be used to effect an event transition from one state to another.
Event	Event name
Condition	Not supported in the release

- Select the Event Group from the drop down
- Select the Event and click Accept
- Click cancel to cancel the operation and close the pop up.
- The event action is seen at the transition

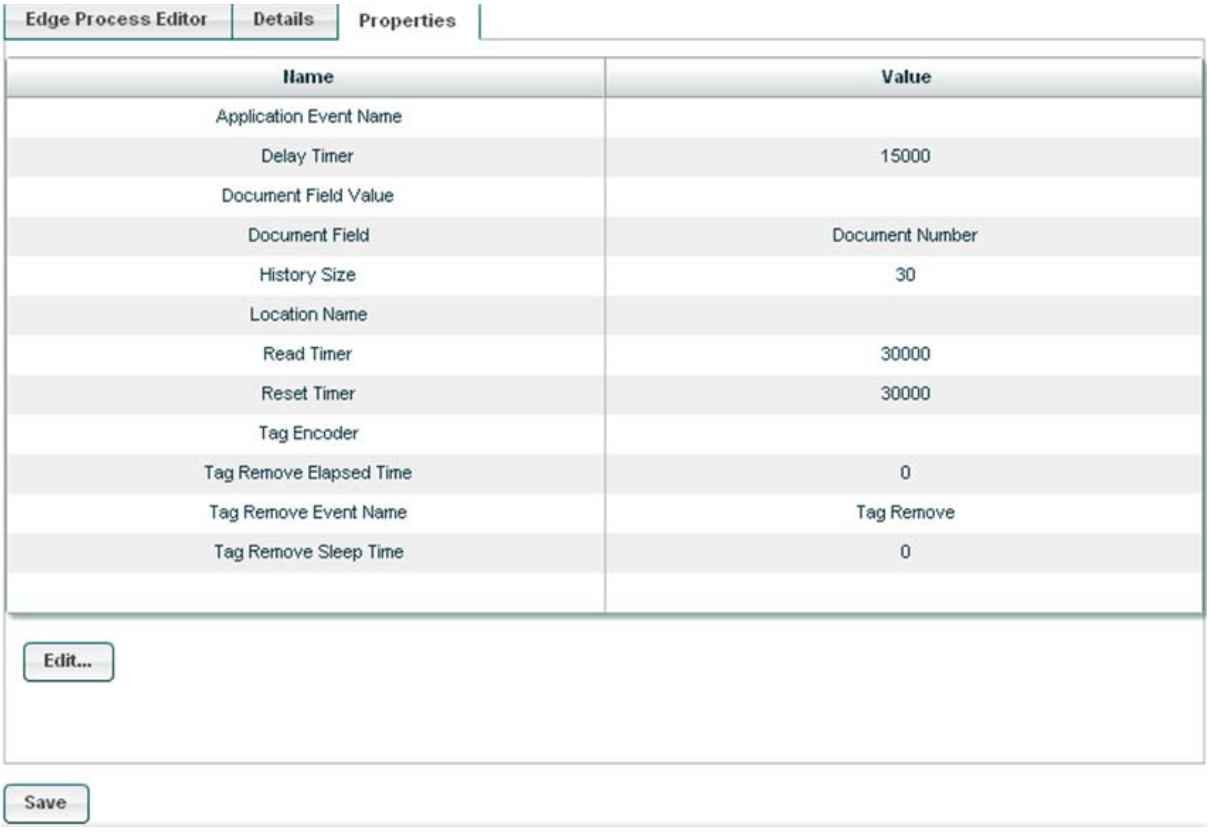




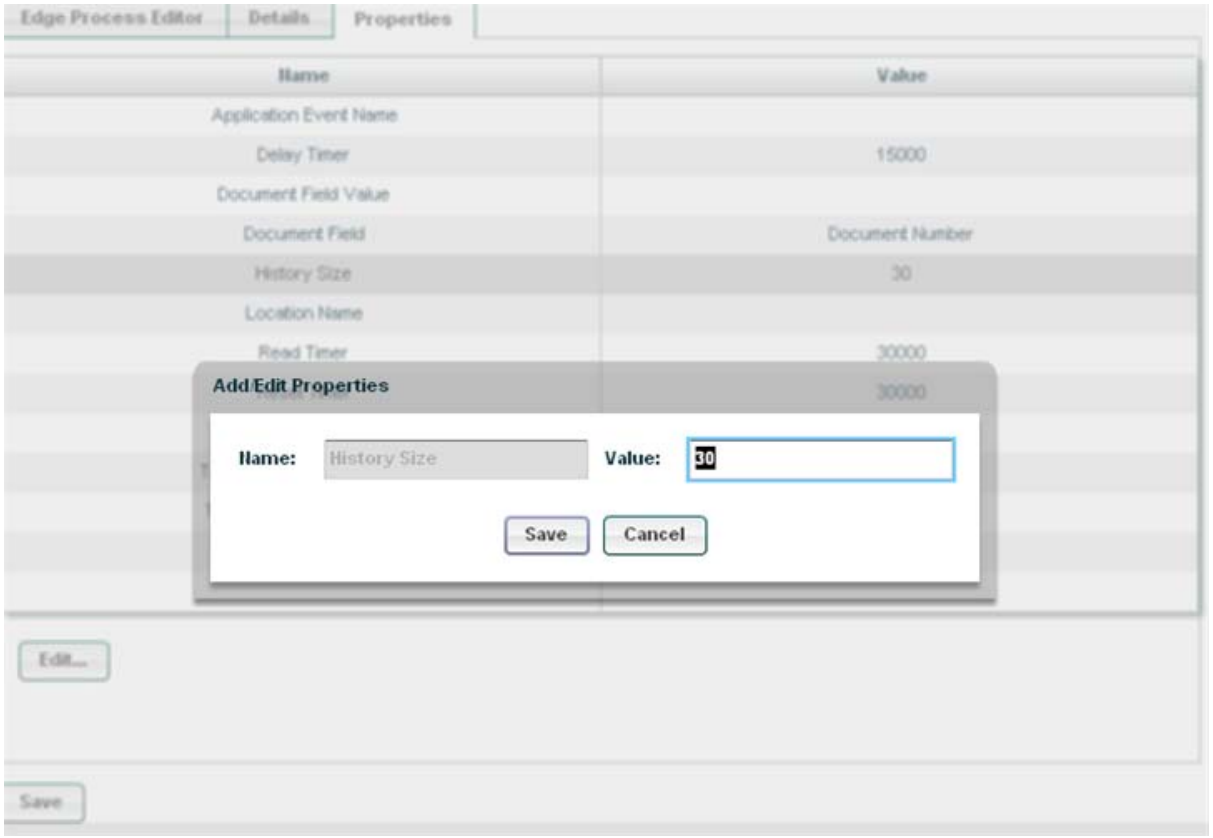
Attribute	Description
Name	Name of the Edge Process. Change the default name. The Edge Process name is unique across all Enterprise Organizations meaning you cannot have the same name in the same Enterprise or other Enterprise organization
Status	Denotes whether the Edge Process is Active or Inactive.
Description	Short notes
Instances	Displays all instances created from the Edge Process

Edge Process Properties
The edge process properties act as the input to the transition actions. The Properties for the Edge Process are defined in the Control Object and if used in the Actions then the value for the property is set here. In the absence of the value the default value set by the Control Object is used by the Action.

Note: For the Properties to show up a control object should have been selected for one of the states.

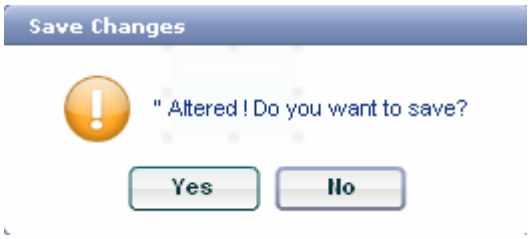


To Edit the property select the property and click on ‘Edit’



Change the value and click ‘Save’ to save the property

- Finally click ‘Save’ to save the Edge Process
- If the Edge Process is not saved and navigated to a different screen the system prompts the user with a warning message.



- Click Yes to Save the Edge Process
- Click No to cancel the changes.

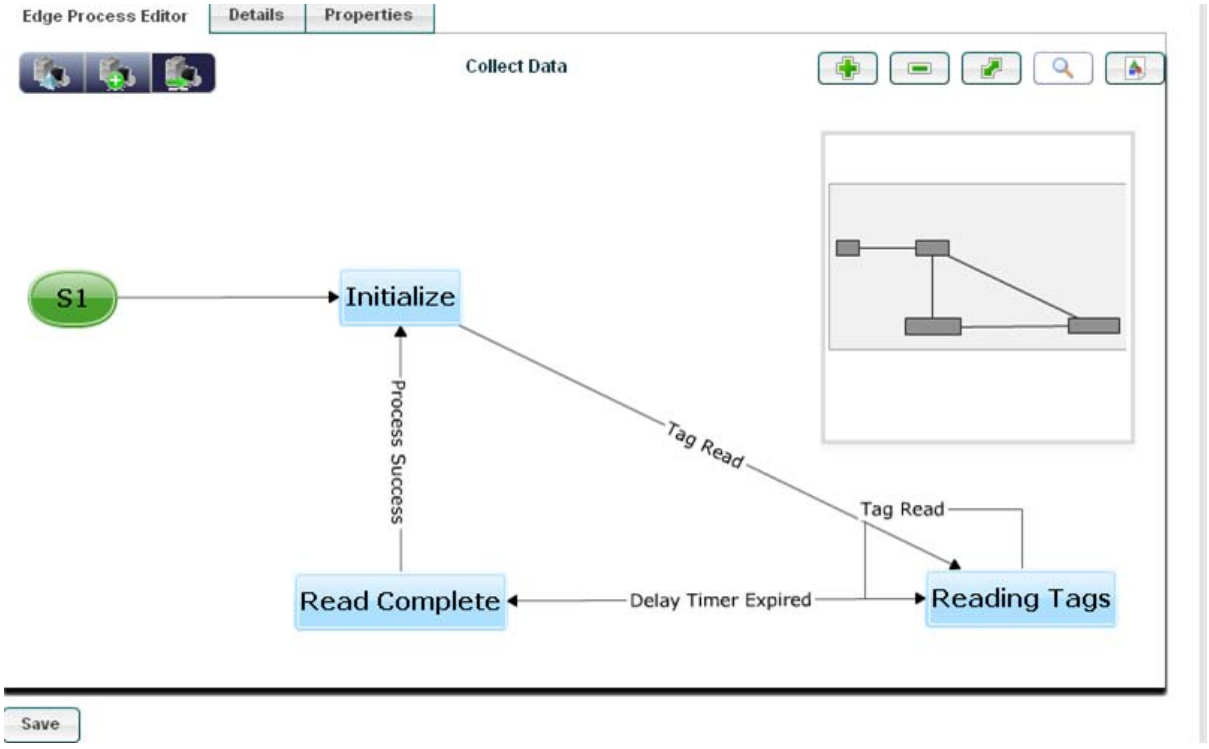
4.7.1 Edit Edge Process

- To edit an Edge Process select the Edge processes
- Update the process according to the instructions given in the previous section
- Click on Save button to update the properties

4.7.2 Modeler Tool bar

The tool bar has useful features that can be used while creating / updating the edge process.

1. To zoom into the edge process. The editor view maximizes as the button is clicked
2. To zoom out of the edge process. The editor view minimizes as the button is clicked
3. Fit Content – Aligns the edge process to the editor window for a better view.
4. Toggle overview – opens a screen on the editor panel. Drag the mouse in the screen to which the editor also moves along with the mouse movements in the screen. Click again to come out of the Toggle mode.



- Full Screen – opens the edge process screen in full screen mode.
- Operations possible on the Edge Process are
- Delete Edge Process
- Select the Edge Process and click on button.

Validate Edge Process

- The Validate Edge Process checks the transitions created and verifies if the transitions are all valid.

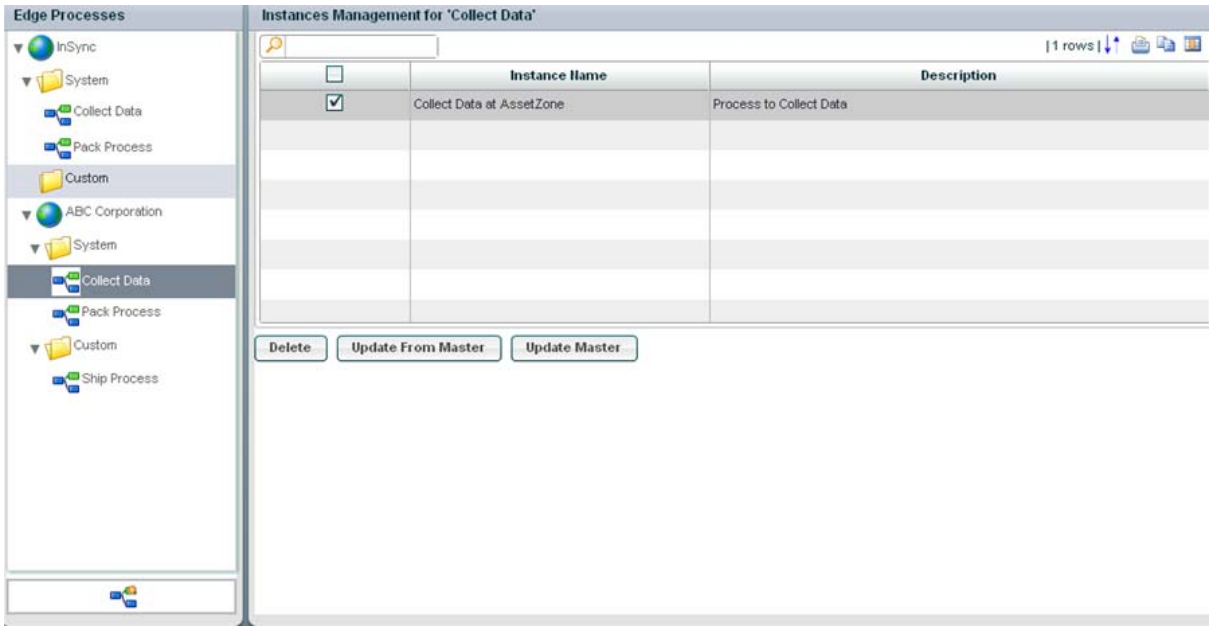
- A Minimum of 2 transitions is required for an Edge Process. Validation fails for a single transition edge process.

Copy Edge Process

- Use the feature to create a copy of an existing Edge Process with a different name. The transitions can be changed on the new edge process.

Manage Instances

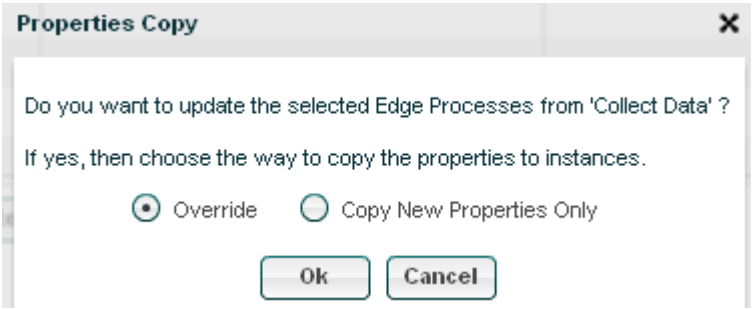
- Manage Instances lists all the Edge Process instances created for a selected Edge Process.



Select the check box against the instance name to

A. Delete: Deletes the selected instance if it is not associated to a process zone.

B. Update from Master – Updates the instances with the Master Edge Process changes


Properties Copy
Do you want to update the selected Edge Processes from 'Collect Data' ?
If yes, then choose the way to copy the properties to instances.
Override
Copy New Properties Only
Ok
Cancel

Select 'Override' to override everything of the instance from the master.
Select 'Copy New Properties Only' to copy only the properties from the master.

C. Update Master – Updates the Master from which the instance has been created with the instance changes.

Note: For System Edge Process at the child enterprise only the Manage Instance action is available while all the actions are available at the Home Enterprise

Note: Edge Processes can be associated to the device and to the Process Zone from the Explorer panel.

4.8 External Device Types

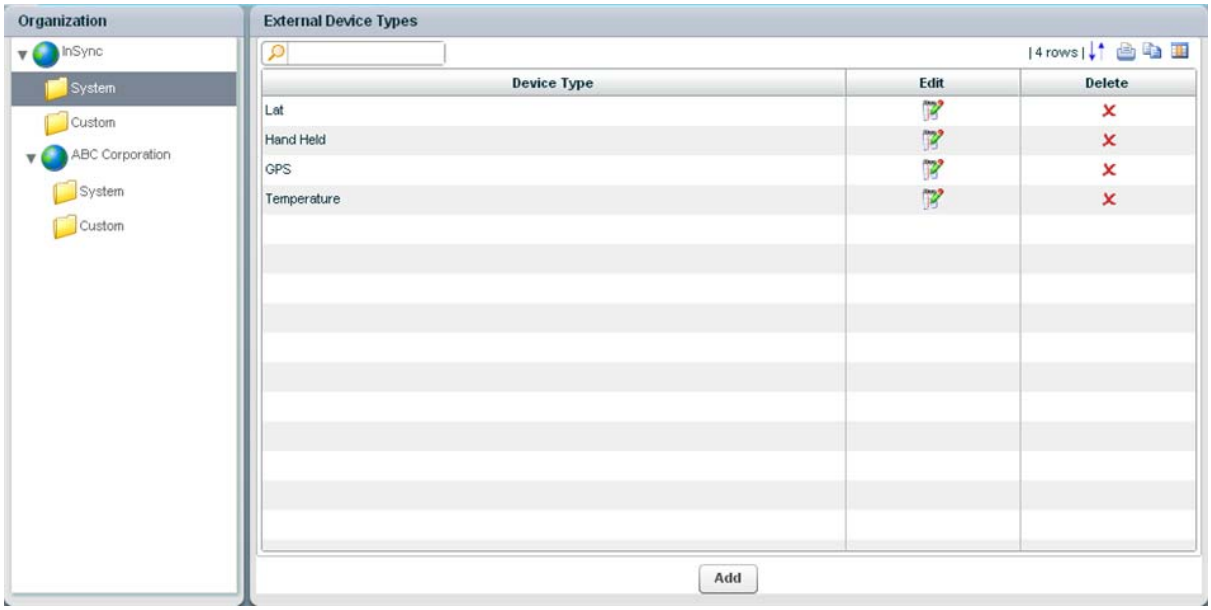
Devices other than fixed RFID readers which are used in communicating with IApp are configured using the external device types and external devices. For such devices to communicate to IApp they have to be registered with IApp.

External Device Types are primarily to differentiate between mobile devices and other devices like GPS and temperature based devices.

The external device types in IApp are listed by Enterprise – System / Custom.

- The System device types are provided by default and are available across all Enterprises.
- Only users with Administrative rights at the Home organization will be able to add or modify the System device types
- Users belonging to an Enterprise organization will be able to add new Custom device types and modify the existing custom device types in that Enterprise only.
- Administrative users at Home Organization will be able to add Custom device types in other Enterprise organizations.

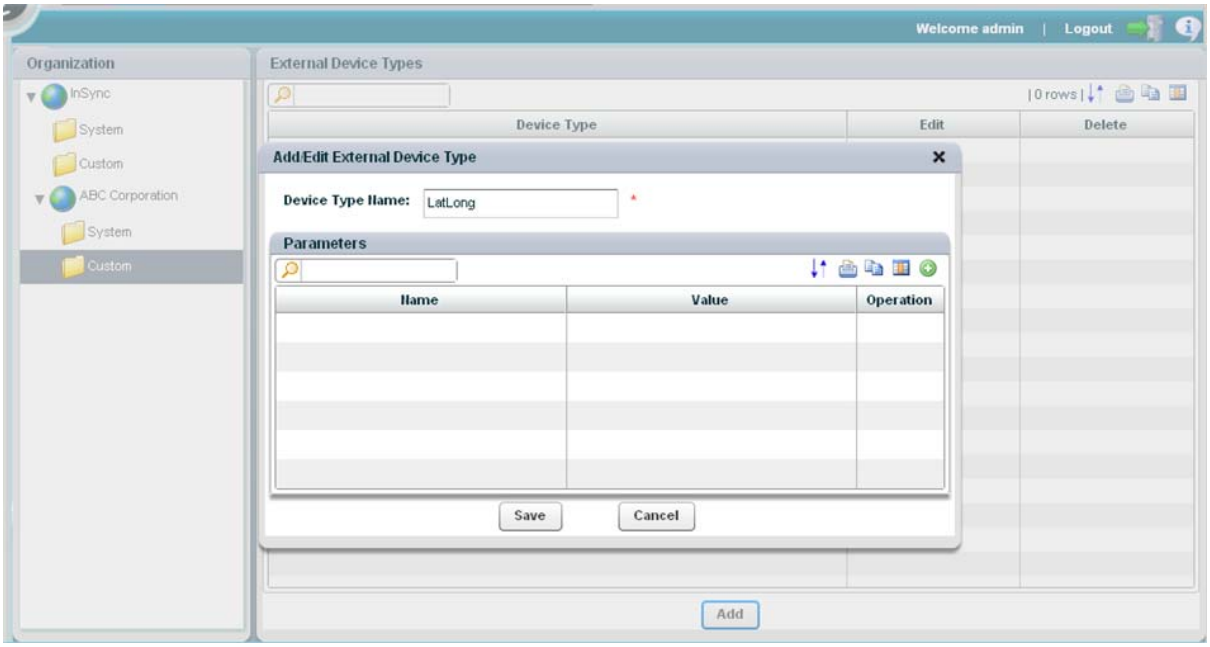
Note: Custom mobile applications should be built and deployed on the handheld devices for it to be able to communicate to IApp and these handheld devices should be registered with IApp for the communication to be succesful



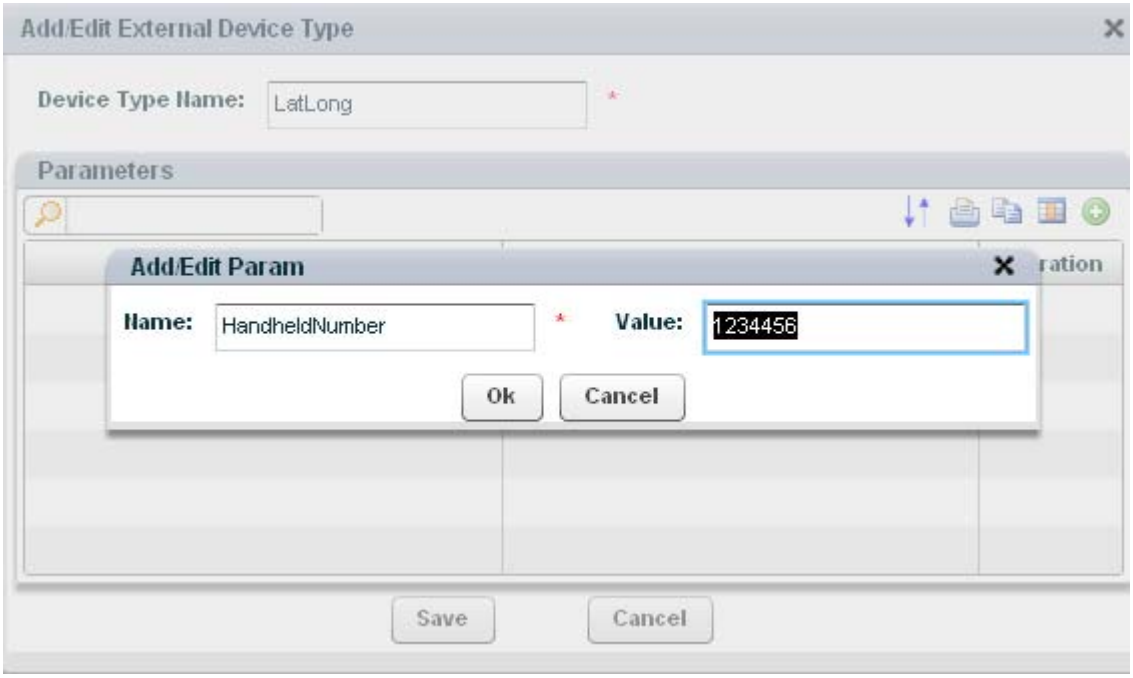
4.8.1 Add Device Type

To create a external device type

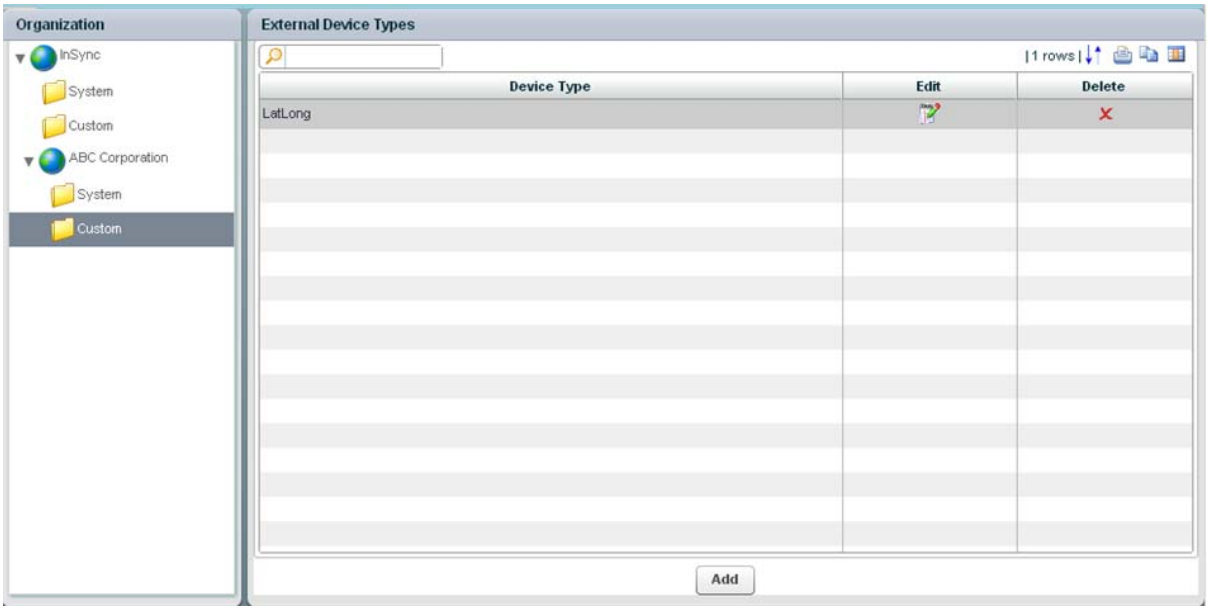
1. Select the Organization
2. Select System or Custom based on the access.
3. Click Add



4. Click on the custom data grid to add parameters. The parameters can be any static property with a value set.



5. Click Ok, Save to save the Mobile Device Type. Click Cancel to cancel the operation.

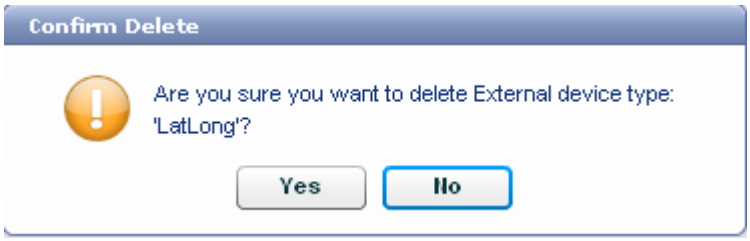


4.8.2 Edit Mobile Device Type

1. Click against the device type to be edited.
2. Click Save to save the changes or cancel to cancel the operation

4.8.3 Delete Mobile Device Type

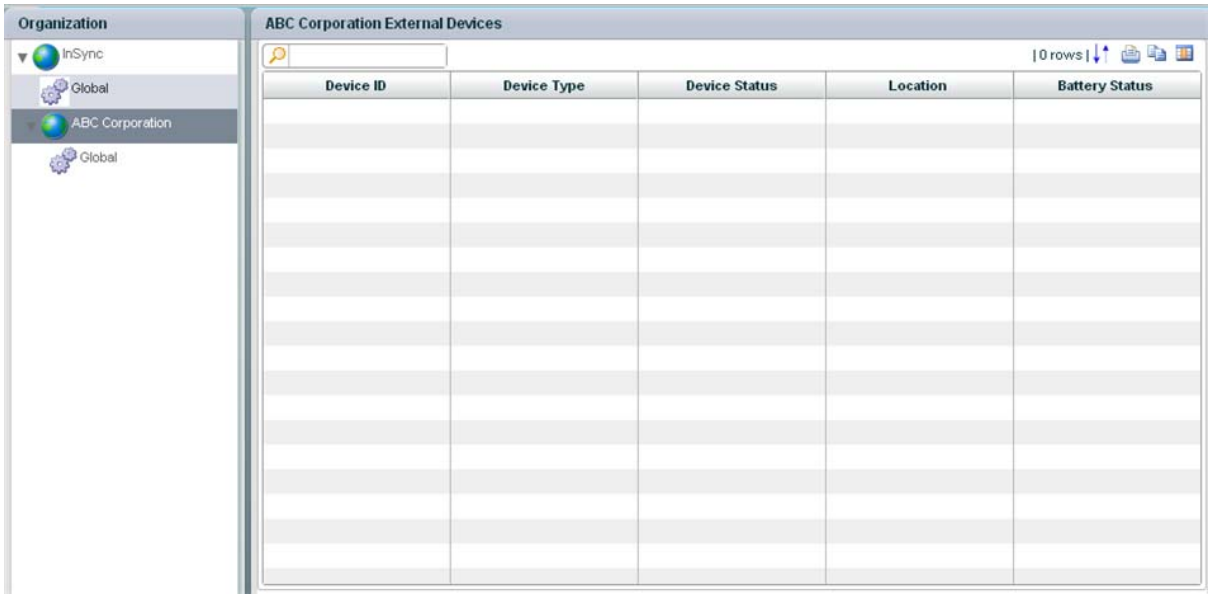
1. Click against the device type to be deleted.



2. Click Yes to confirm the deletion, No to cancel the operation

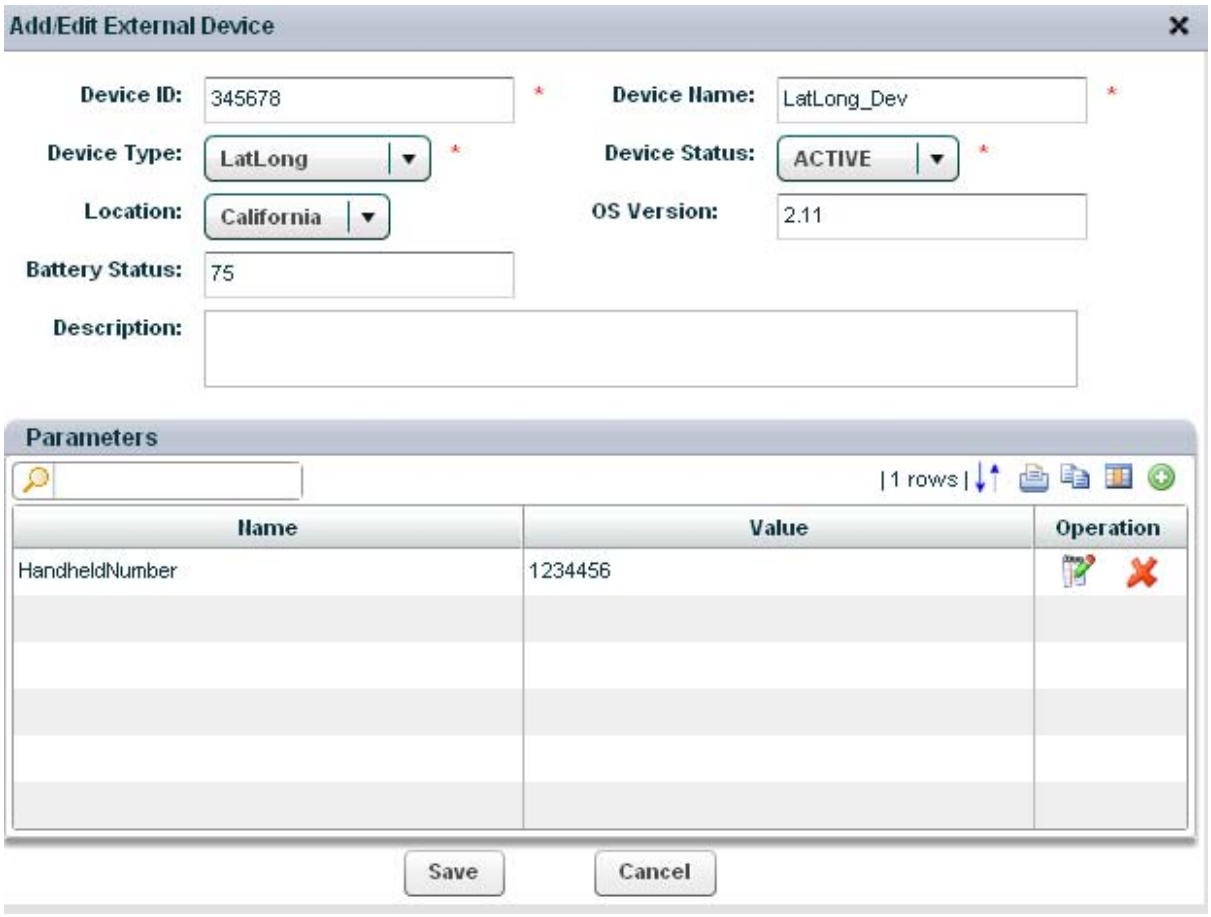
4.9 External Devices

Multiple devices can be added for a external device type. The external devices page allows for registering such devices without which the device will be unable to communicate to IApp.



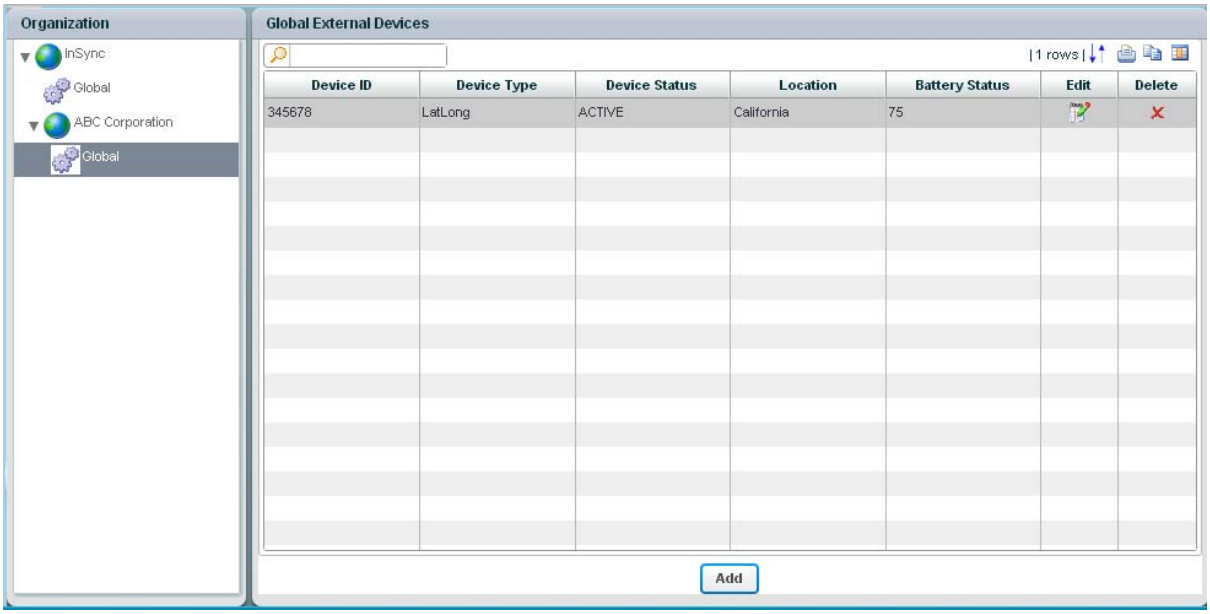
4.9.1 Add external Device

1. Select Default zone in the enterprise where the device has to be added.
2. Click on Add to add a new device



Attribute	Description
Device ID	Is the actual serial number of the device provided by the manufacturer.
Device Name	Name of the external device
Device Status	Denotes whether the device is Active or Inactive.
Device Type	Type of the mobile device type for which the device is getting created
Location	the location created as part of the location hierarchy
Battery Status	Status of the battery.
OS version	Version of the OS on the handheld

3. Click *Save* to save the device.



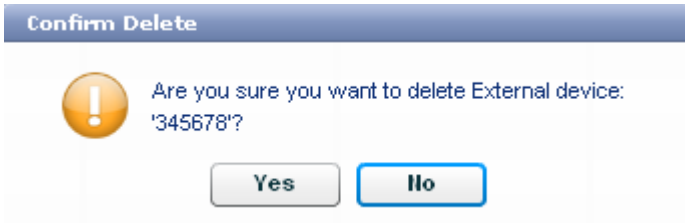
4. Click C*ancel* to cancel the operation

4.9.2 Edit Devices

1. Click against the device to be edited.
2. Click Save to save the changes or cancel to cancel the operation

4.9.3 Delete Mobile Device Type

1. Click against the device to be deleted.



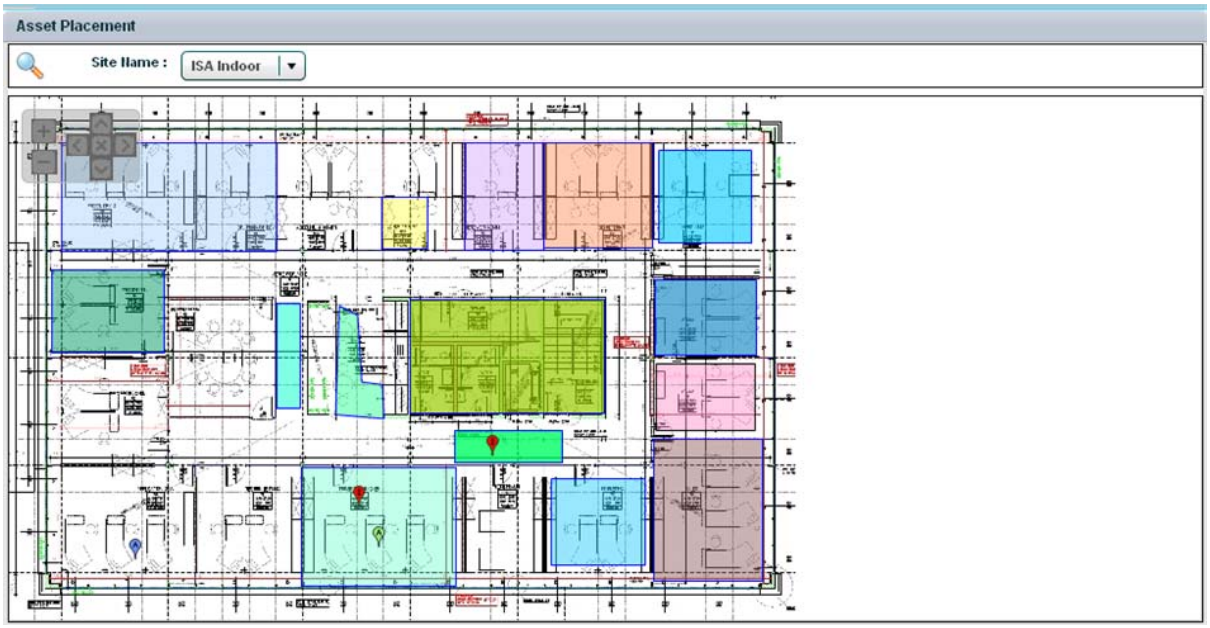
2. Click Yes to confirm the deletion, No to cancel the operation

Note:

1. All external device communication with IApp have to be registered with IApp.
2. Appropriate license files are needed to be able to create the devices without which the creation of the external devices will fail.

4.10 Asset Placement

The asset placement screen is used when an asset has to be placed manually in a region or asset co-ordinates have to be corrected if the asset has to be moved manually from its initial location.



- The site name drop down displays the RTLS location configured in the Location hierarchy.

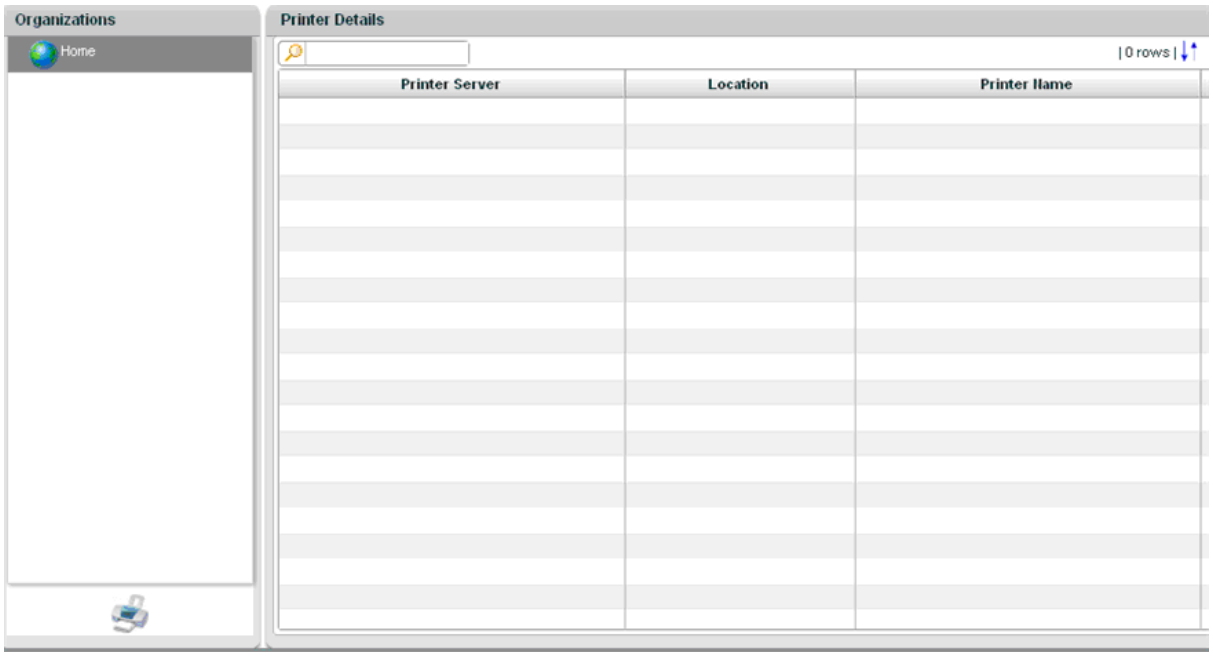
- To change the asset placement select the asset on the viewer and drag/place it at a different place on the viewer.

- The small panel with the X, Y, Z postions shows the latest co-ordinates of the asset.

- Click *Update* to update the new asset placement.

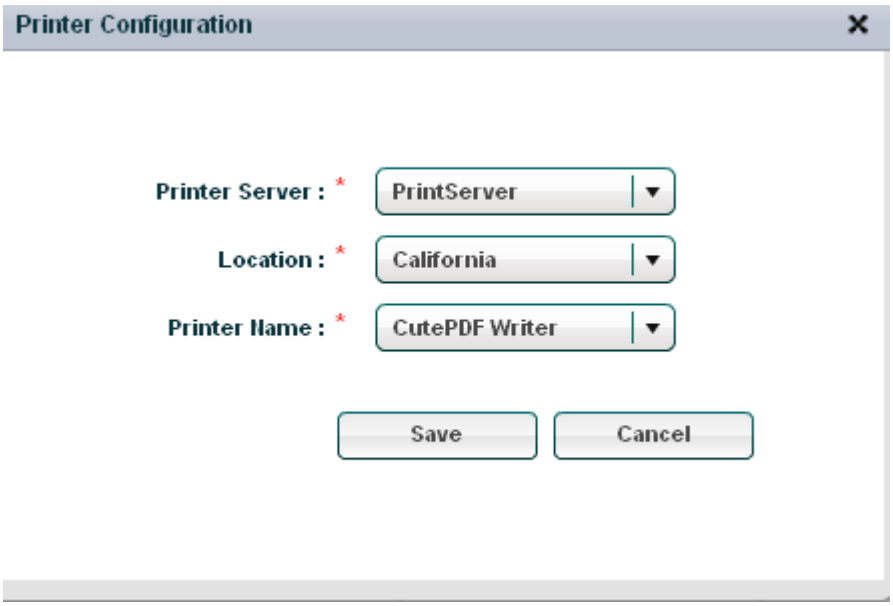
4.11 Printers

Printers available (drivers installed) with the Print server instance registered in the instance registry are configured using the Printer section. This allows for selecting the network printer available with the print server instance. More than one printer can be configured on a single print server instance.



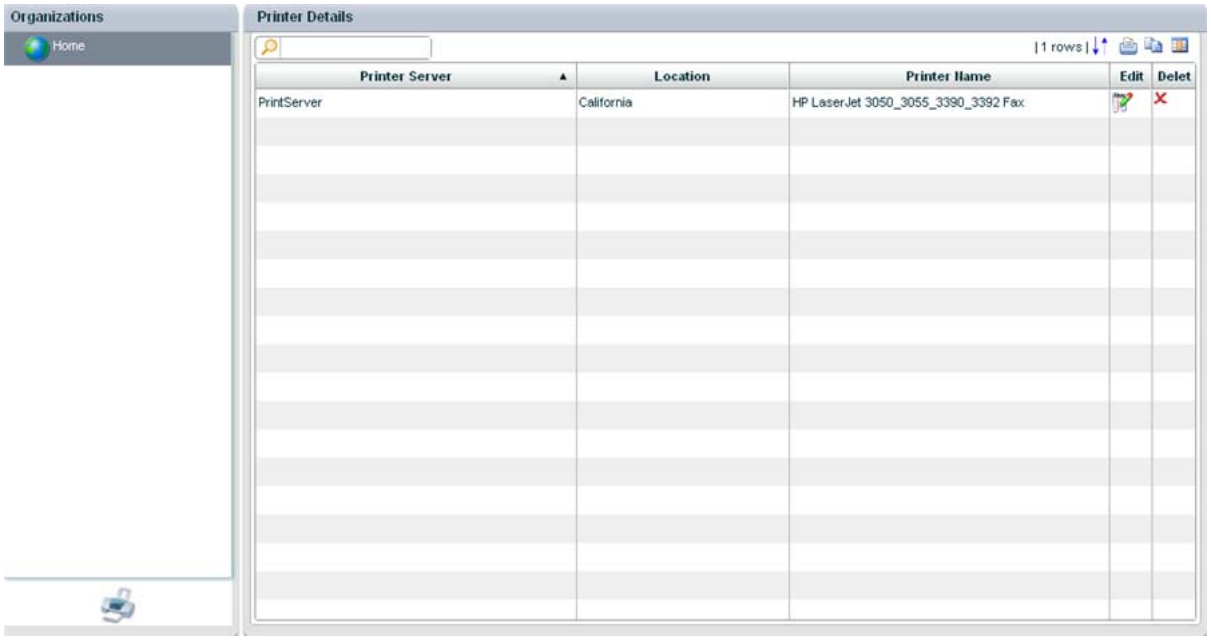
4.11.1 Add Printer

1. Select the Enterprise organization
2. Click



Attribute	Description
Print Server	Lists all the printer server instances registered for the Enterprise using the instance registry
Location	is the location from the Enterprise Org location hierarchy
Printer Name	Lists all the printer drivers registered on the machine hosting the print server instance.

3. Click Save to commit the changes

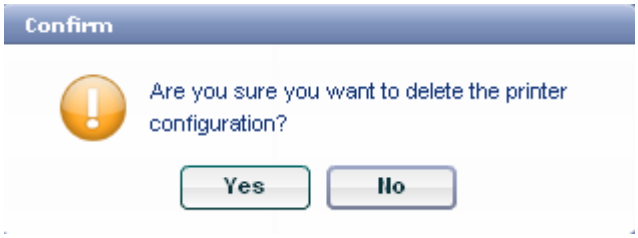


4.11.2 Edit Printer

1. Click against the printer to be edited.
2. Click Save to save the changes or cancel to cancel the operation

4.11.3 Delete Printer

1. Click against the printer to be deleted

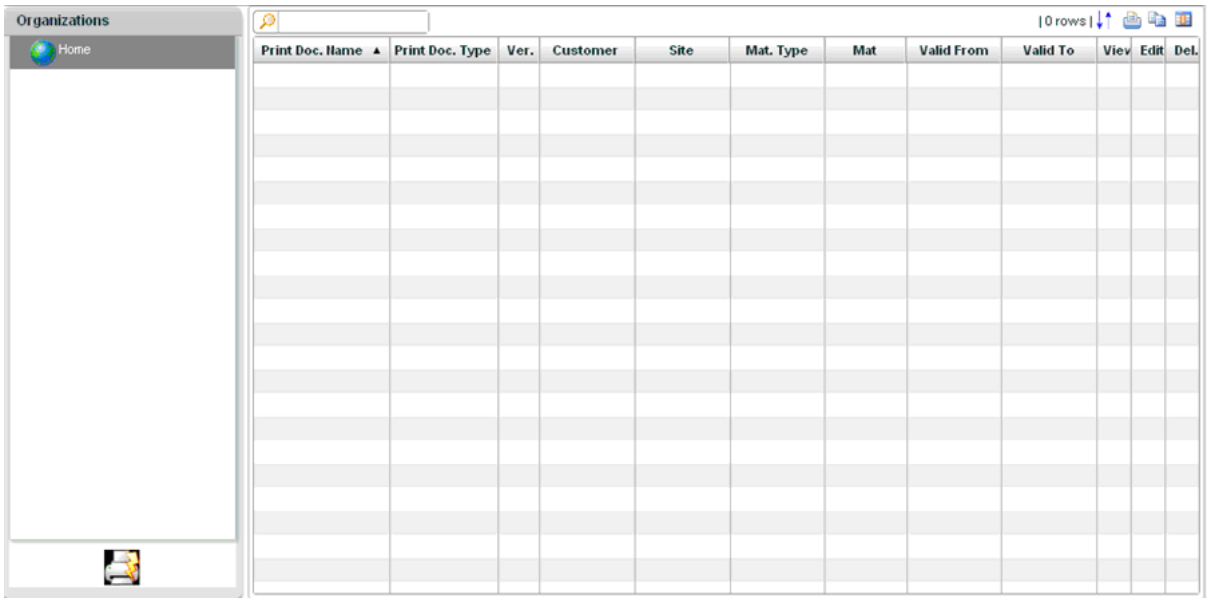


2. Click Yes to confirm the deletion, No to cancel the operation

4.12 Printer Documents

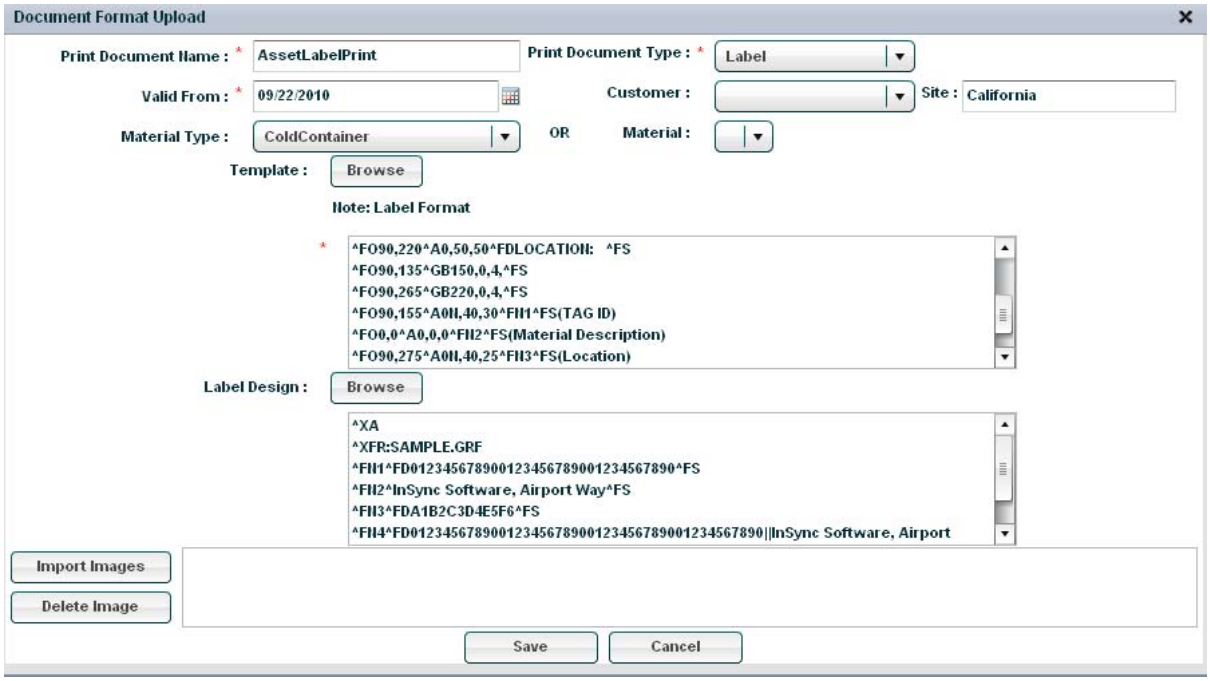
The Printer document section allows configuring the final print format that would be sent to the printer and the filtering rules surrounding it.

Currently iApp print capability supports printing PDF, text and label formats, the templates for which can be configured.



4.12.1 Add a Document Format

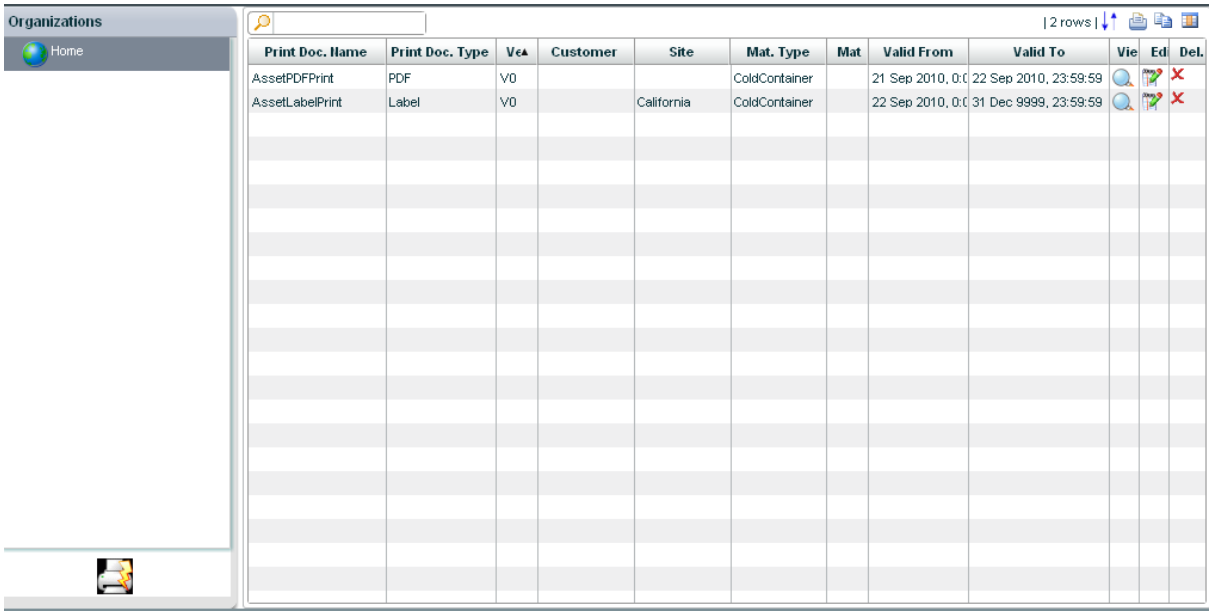
1. Select the Enterprise organization for which the document format has to be created.
2. Click



Attribute	Description
Print Document Name	Name for the print document. Multiple documents with the same name can be created with varying formats. The document to be picked for printing depends on the criteria set while creating the document. Refer to the Document Search Criteria discussed in the later sections.
Print Document Type	Type of document to be printed, currently supports Label, PDF and text.
Valid From	Date from which the document print gets active and would be in force. **Note:** Till a different version of the document gets created the valid till date is set to 31 Dec 9999. Refer to the document versioning listed below.
Customer	Enterprises of type Customer are listed here
Site	Location name
Material Type	Material Types from the list available
Material	Materials from the available list.
Template	The label template designed by the subject expert. The language used to create would vary based on the printer specifications and the document type. For documents of type PDF and text the template will be a XSL style sheet matching to the event.
Label Design	The data feed for the place holders in the template created. This again follows the language supported by the printer. **Note:** • It is not necessary that the data comes in from the design. This can be part of the document template as well based on the printers. Some printers require the template to be sent first and

	then the label data. In such cases both the template and the label data would be required. • The label design is disabled for documents of Type PDF and text.
Import Images / Delete Image	A sample image to be saved. This has nothing to do with the template and the design and is only for illustration.

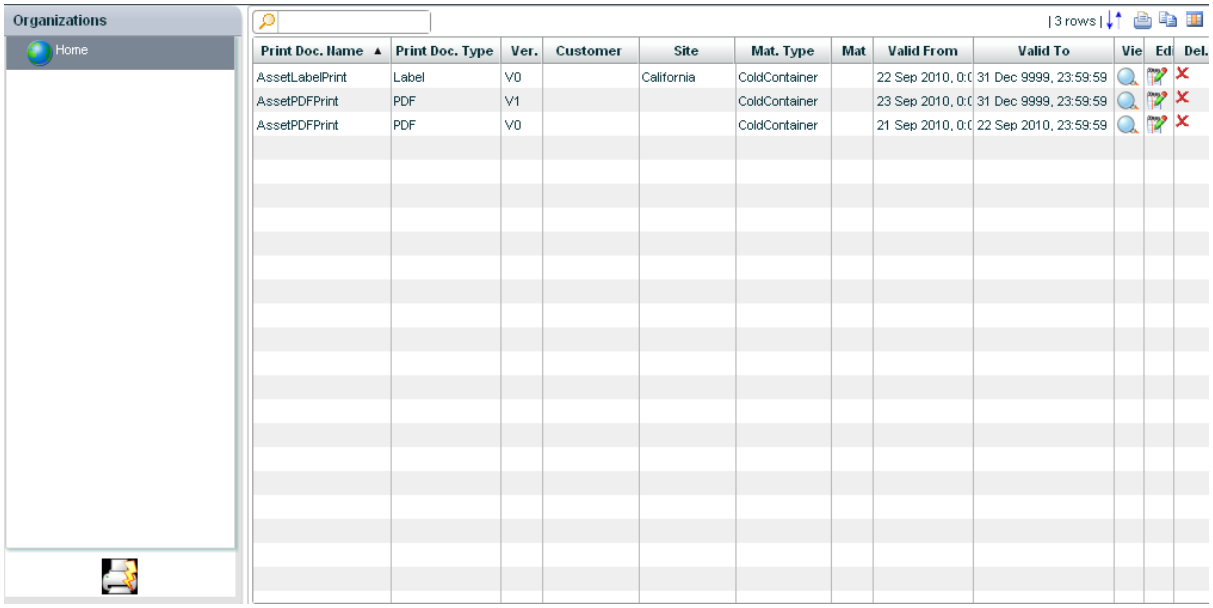
3. Click Save to save the document
4. Click cancel to cancel the changes.



4.12.2 Document Versioning:

iApp Printer feature supports print document versioning and follow the set rules.

- Versioning happens for Active documents only. The print document should be Active meaning that the current date falls between the valid from and valid to date of the document.
- The first version of the document created is set to V0.
- Editing an Active document will create a new version V1 with the valid to date of version V0 set to a day less than the valid from date of the new version. While editing the document it is mandatory to change the valid from date and should be greater than the current date.



- When a new version of document is created, the system does not allow the older version to be edited but can be viewed.
- An active document in force and older versions of the document cannot be deleted.
- Higher Document versions set to be active in the future can be deleted or edited for changes.
- Creating new versions of the document is possible only on the higher Active version. The system does not allow versioning on an Active document if a higher version is available even if it is set to a future valid from date.
- If for some reason a new version V1 gets created with the valid from date set to a future date and gets deleted then the V0 version valid to date does not get reset to the year 9999 and will remain to the earlier date that was set when the V1 version got created.
- Multiple documents with the same name can be created with varying formats. The document to be picked for printing depends on the criteria set while creating the document. Refer to the Document Search Criteria discussed in the later sections.

4.12.3 Label Templates and Design

- The label template is the layout of the document that would be set to print with place holders for the data that would come in as input in the form of label design to the template.
- The label design is the data input to the label template.
- The templates and the data design are scripted in the language supported by the printers in use.
- The data design can be scripted as part of the template itself for certain printers and not as a separate script.
- For certain printers it is mandatory that the template and the data design are scripted separately with the template being sent first and then the label design.
- Based on the document type (Label, PDF, text) the template and the design will vary.
- For documents of type Label the template and label design (not mandatory) are required
- For documents of type PDF and Text the template is only required.

- For documents of type PDF the template should be an XSL using Formatting objects matching to the incoming event format.
- For document of type Text the incoming event is formatted as below for which a valid XSL has to be written.

  <?xml version=1.0?>
  <PrintData>
  <Data></Data>
  </PrintData>

- If the XSL is not provided for documents of type TEXT then the built in default XSL is used for the conversion process.

4.12.4 Document Search Criteria

For documents with the same name the system follows the search criteria listed below based on the different attributes and the combinations.

The attributes shown are for Customer C1, Material M1, Material type MT1 and Site S1 for a document named AssetPrintLabel.

Table 1 shown is an illustration on data stored in the database with a unique ID and table 2 shows the document ID picked based on the attribute and attribute combination.

Table 1: sample snapshot of the data stored for document name AssetPrintLabel

ID	Document Name	Customer	Material	Material Type	Site
1	AssetPrintLabel	C1	M1		S1
2	AssetPrintLabel	C1	M1		
3	AssetPrintLabel	C1	MT1		S1
4	AssetPrintLabel	C1	MT1		
5	AssetPrintLabel	C1			
6	AssetPrintLabel		M1		S1
7	AssetPrintLabel		M1		
8	AssetPrintLabel			MT1	S1
9	AssetPrintLabel			MT1	
10	AssetPrintLabel				S1
11	AssetPrintLabel				

Table 2: The table shows the document that gets picked from the documents available in the system shown in table 1.

Document Name	Customer	Material	Material Type	Site	Document Picked
AssetPrintLabel	C1	M1		S1	1
AssetPrintLabel	C1	M1			2
AssetPrintLabel	C1	MT1		S1	3
AssetPrintLabel	C1	MT1			4
AssetPrintLabel	C1				5
AssetPrintLabel		M1		S1	6
AssetPrintLabel		M1			7
AssetPrintLabel			MT1	S1	8
AssetPrintLabel			MT1		9
AssetPrintLabel				S1	10
AssetPrintLabel					11
	Error				

Note: The event for creating the print document might come in from the Auto Print Handler if configured in the system or the Print SDK can be used to raise the event and appropriate properties set. Refer to iApp Event Handler section for information on the Auto Print Handler.

4.12.5 Brief Steps to print a document using a print server

- Create the printer configuration
- Create a print document with the required template if applicable.
- Configure the Auto print handler with the right handler properties which should match to the printer/print document configuration

4.12.6 Printer SDK

To use the printer infrastructure from a custom class in sensornet

- Use the class

  com.insync.edgeware.sensornet.printer.DocumentPrinter

  The following methods are available

  - public void printDocument(PrintRequestVO printRequestVO) throws Exception
    - The method takes care of discovering the print server and sending the request to the print server.
    - The method can be used to print documents of type TXT, PDF and lbl

  - public byte[] getPrintText(String text, String xslt) throws Exception
    - converts text into printable byte data
    - Uses the xslt if provided for the conversion else it will use the default xslt.
    - If xslt is provided the text parameter should be in xml. If it is simple text then the xslt you provide should be able to parse the xml of the format

```
<?xml version=1.0?>
<PrintData>
<Data></Data>
</PrintData>
```

- **Example using an xslt**

  - Assuming the print server is created with the following configuration

    Print Server = PrintServer
    Location = Hyderabad
    Printer Name = “HP Universal Printing PCL 6”

  - The printer document is created with the following parameters

    Print Document Name = “Doc1”
    Print Document Type = “Text”
    Valid From = “03-03-2011” (If this document to become active this date should be less than current date)

    Customer, Site, Material Type, Material are left blank.

And template is

- - - -

```
<fo:block>
        <xsl:copy-of select="$value-of-template_0E13F868"/>
</fo:block>
</xsl:when>
<xsl:otherwise>
<fo:inline>
        <xsl:copy-of select="$value-of-template_0E13F868"/>
</fo:inline>
</xsl:otherwise>
</xsl:choose>
                                                        </xsl:for-each>
                                                        <fo:block/>
                                                        <fo:inline>
<xsl:text>Location : </xsl:text>
                                                        </fo:inline>
                                                        <xsl:for-each
select="Loc">
<xsl:variable name="value-of-template_0E13FA90">
<xsl:apply-templates/>
</xsl:variable>
<xsl:choose>
<xsl:when test="contains(string(value-of-template_0E13FA90),'&#x2029;')">
<fo:block>
        <xsl:copy-of select="$value-of-template_0E13FA90"/>
</fo:block>
</xsl:when>
<xsl:otherwise>
<fo:inline>
        <xsl:copy-of select="$value-of-template_0E13FA90"/>
</fo:inline>
</xsl:otherwise>
</xsl:choose>
                                                        </xsl:for-each>
```

```
                                                                    <fo:block/>
                                                                    <fo:inline>
            <xsl:text>Time : </xsl:text>
                                                                    </fo:inline>
                                                                    <xsl:for-each
select="Time">
            <xsl:variable name="value-of-template_0E13F9D8">
            <xsl:apply-templates/>
            </xsl:variable>
            <xsl:choose>
            <xsl:when test="contains(string(value-of-template_0E13F9D8),'&#x2029;')">
            <fo:block>
                  <xsl:copy-of select="$value-of-template_0E13F9D8"/>
            </fo:block>
            </xsl:when>
            <xsl:otherwise>
            <fo:inline>
                  <xsl:copy-of select="$value-of-template_0E13F9D8"/>
            </fo:inline>
            </xsl:otherwise>
            </xsl:choose>
                                                                    </xsl:for-each>
                                                                    <fo:block/>
                                                                    <fo:inline>
            <xsl:text>Device : </xsl:text>
                                                                    </fo:inline>
                                                                    <xsl:for-each
select="Device">
            <xsl:variable name="value-of-template_0E13FD70">
            <xsl:apply-templates/>
            </xsl:variable>
            <xsl:choose>
            <xsl:when test="contains(string(value-of-template_0E13FD70),'&#x2029;')">
            <fo:block>
                  <xsl:copy-of select="$value-of-template_0E13FD70"/>
```

```
</fo:block>

</xsl:when>

<xsl:otherwise>

<fo:inline>

        <xsl:copy-of select="$value-of-template_0E13FD70"/>

</fo:inline>

</xsl:otherwise>

</xsl:choose>
                                                                </xsl:for-each>
                                                                <fo:block/>
                                                                <fo:inline>

<xsl:text>Zone : </xsl:text>
                                                                </fo:inline>
                                                                <xsl:for-each
select="Zone">

<xsl:variable name="value-of-template_0E13FCB8">

<xsl:apply-templates/>

</xsl:variable>

<xsl:choose>

<xsl:when test="contains(string(value-of-template_0E13FCB8),'&#x2029;')">

<fo:block>

        <xsl:copy-of select="$value-of-template_0E13FCB8"/>

</fo:block>

</xsl:when>

<xsl:otherwise>

<fo:inline>

        <xsl:copy-of select="$value-of-template_0E13FCB8"/>

</fo:inline>

</xsl:otherwise>

</xsl:choose>
                                                                </xsl:for-each>
                                                                <fo:block/>
                                                                <fo:inline>

<xsl:text>Organization : </xsl:text>
```

```
                                                                    </fo:inline>
                                                                    <xsl:for-each
select="Org">
        <xsl:variable name="value-of-template_0E13FC00">
        <xsl:apply-templates/>
        </xsl:variable>
        <xsl:choose>
        <xsl:when test="contains(string(value-of-template_0E13FC00),'&#x2029;')">
        <fo:block>
                <xsl:copy-of select="$value-of-template_0E13FC00"/>
        </fo:block>
        </xsl:when>
        <xsl:otherwise>
        <fo:inline>
                <xsl:copy-of select="$value-of-template_0E13FC00"/>
        </fo:inline>
        </xsl:otherwise>
        </xsl:choose>
                                                                    </xsl:for-each>
                                                                    <fo:block/>
                                                                    <fo:block>
                                                                        <fo:leader
leader-pattern="space"/>
                                                                    </fo:block>
                                                                    <fo:block>
                                                                        <fo:leader
leader-pattern="space"/>
                                                                    </fo:block>
                                                                    <fo:inline>
        <xsl:text>Tag Data</xsl:text>
                                                                    </fo:inline>
                                                                    <fo:block/>
                                                                    <fo:inline>
        <xsl:text>_______</xsl:text>
                                                                    </fo:inline>
                                                                    <fo:block/>
                                                                    <fo:block>
                                                                        <fo:leader
leader-pattern="space"/>
                                                                    </fo:block>
                                                                    <xsl:for-each
select="Tag">
```

```
                                                                    <fo:block>
        <fo:leader leader-pattern="space"/>
                                                                    </fo:block>
                                                                    <fo:inline>
        <xsl:text>Tag ID : </xsl:text>
                                                                    </fo:inline>
                                                                    <xsl:for-
each select="ID">
        <xsl:variable name="value-of-template_0E13FB48">
        <xsl:apply-templates/>
        </xsl:variable>
        <xsl:choose>
        <xsl:when test="contains(string(value-of-template_0E13FB48),'&#x2029;')">
                <fo:block>
                        <xsl:copy-of select="$value-of-template_0E13FB48"/>
                </fo:block>
        </xsl:when>
        <xsl:otherwise>
                <fo:inline>
                        <xsl:copy-of select="$value-of-template_0E13FB48"/>
                </fo:inline>
        </xsl:otherwise>
        </xsl:choose>
                                                                    </xsl:for-
each>
                                                                </xsl:for-each>
                                                                <fo:block/>
                                                                <fo:inline>
        <xsl:text>Antenna : </xsl:text>
                                                                </fo:inline>
                                                                <xsl:for-each
select="Tag">
                                                                    <xsl:for-
each select="Antenna">
        <xsl:variable name="value-of-template_0E13FF98">
        <xsl:apply-templates/>
        </xsl:variable>
```

```
<xsl:choose>
<xsl:when test="contains(string(value-of-template_0E13FF98),'&#x2029;')">
<fo:block>
<xsl:copy-of select="$value-of-template_0E13FF98"/>
</fo:block>
</xsl:when>
<xsl:otherwise>
<fo:inline>
<xsl:copy-of select="$value-of-template_0E13FF98"/>
</fo:inline>
</xsl:otherwise>
</xsl:choose>
                                                                    </xsl:for-
each>
                                                                    <fo:block/>
                                                                    <fo:inline>
<xsl:text>TagRead time : </xsl:text>
                                                                    </fo:inline>
                                                                    <xsl:for-
each select="TagReadTime">
<xsl:variable name="value-of-template_0E140050">
<xsl:apply-templates/>
</xsl:variable>
<xsl:choose>
<xsl:when test="contains(string(value-of-template_0E140050),'&#x2029;')">
<fo:block>
<xsl:copy-of select="$value-of-template_0E140050"/>
</fo:block>
</xsl:when>
<xsl:otherwise>
<fo:inline>
<xsl:copy-of select="$value-of-template_0E140050"/>
</fo:inline>
```

```
		</xsl:otherwise>

		</xsl:choose>
                                                                                </xsl:for-
each>
                                                                                <fo:block/>
                                                                        </xsl:for-each>
                                                                        <fo:block/>
                                                                        <fo:block>
                                                                                <fo:leader
leader-pattern="space"/>
                                                                        </fo:block>
                                                                        <fo:inline>

		<xsl:text>End of Event Data.</xsl:text>
                                                                        </fo:inline>
                                                                        <fo:block/>
                                                                </xsl:for-each>
                                                                <fo:block/>
                                                        </xsl:for-each>
                                                </fo:block>
                                                <fo:block id="SV_RefID_PageTotal"/>
                                        </fo:flow>
                                </fo:page-sequence>
                        </fo:root>
		</xsl:template>
		<xsl:template name="double-backslash">
                        <xsl:param name="text"/>
                        <xsl:param name="text-length"/>
                        <xsl:variable name="text-after-bs" select="substring-after($text, '\')"/>
                        <xsl:variable name="text-after-bs-length" select="string-length($text-
after-bs)"/>
                        <xsl:choose>
                                <xsl:when test="$text-after-bs-length = 0">
                                        <xsl:choose>
                                                <xsl:when test="substring($text, $text-
length) = '\'">
                                                        <xsl:value-of
select="concat(substring($text,1,$text-length - 1), '\\')"/>
                                                </xsl:when>
                                                <xsl:otherwise>
                                                        <xsl:value-of select="$text"/>
                                                </xsl:otherwise>
                                        </xsl:choose>
                                </xsl:when>
                                <xsl:otherwise>
                                        <xsl:value-of select="concat(substring($text,1,$text-
length - $text-after-bs-length - 1), '\\')"/>
                                        <xsl:call-template name="double-backslash">
                                                <xsl:with-param name="text" select="$text-
after-bs"/>
                                                <xsl:with-param name="text-length"
select="$text-after-bs-length"/>
                                        </xsl:call-template>
                                </xsl:otherwise>
                        </xsl:choose>
		</xsl:template>
		<xsl:template name="altova:MakeValueAbsoluteIfPixels">
```

```
		<xsl:param name="sValue"/>
		<xsl:variable name="sBeforePx" select="substring-before($sValue,
'px')"/>
		<xsl:choose>
			<xsl:when test="$sBeforePx">
				<xsl:variable name="nLengthOfInteger">
					<xsl:call-template
name="altova:GetCharCountOfIntegerAtEndOfString">
						<xsl:with-param name="sText"
select="$sBeforePx"/>
					</xsl:call-template>
				</xsl:variable>
				<xsl:variable name="nPosOfInteger" select="string-
length($sBeforePx) - $nLengthOfInteger + 1"/>
				<xsl:variable name="nValuePx"
select="substring($sBeforePx, $nPosOfInteger)"/>
				<xsl:variable name="nValueIn"
select="number($nValuePx) div number($altova:PxPerIn)"/>
				<xsl:variable name="nLengthBeforeInteger"
select="string-length($sBeforePx) - $nLengthOfInteger"/>
				<xsl:variable name="sRest">
					<xsl:call-template
name="altova:MakeValueAbsoluteIfPixels">
						<xsl:with-param name="sValue"
select="substring-after($sValue, 'px')"/>
					</xsl:call-template>
				</xsl:variable>
				<xsl:value-of select="concat(substring($sBeforePx,
1, $nLengthBeforeInteger), string($nValueIn), 'in', $sRest)"/>
			</xsl:when>
			<xsl:otherwise>
				<xsl:value-of select="$sValue"/>
			</xsl:otherwise>
		</xsl:choose>
	</xsl:template>
	<xsl:template name="altova:GetCharCountOfIntegerAtEndOfString">
		<xsl:param name="sText"/>
		<xsl:variable name="sLen" select="string-length($sText)"/>
		<xsl:variable name="cLast" select="substring($sText, $sLen)"/>
		<xsl:choose>
			<xsl:when test="number($cLast) &gt;= 0 and number($cLast)
&lt;= 9">
				<xsl:variable name="nResultOfRest">
					<xsl:call-template
name="altova:GetCharCountOfIntegerAtEndOfString">
						<xsl:with-param name="sText"
select="substring($sText, 1, $sLen - 1)"/>
					</xsl:call-template>
				</xsl:variable>
				<xsl:value-of select="$nResultOfRest + 1"/>
			</xsl:when>
			<xsl:otherwise>
				<xsl:text>0</xsl:text>
			</xsl:otherwise>
		</xsl:choose>
	</xsl:template>
</xsl:stylesheet>
**********
```

- Create object of type com.insync.edgeware.sensornet.printer.PrintRequestVO class

****

```
com.insync.edgeware.sensornet.printer.PrintRequestVO printRequest = new com.insync.edgeware.sensornet.printer.PrintRequestVO();
printRequestVO.setSaasOrgKey(new java.math.BigDecimal(1)); //organization key.

String docToPrint = "<?xml version="1.0" encoding="UTF-8"?><event_data><Event>Tag Read</Event><Loc>ExitDoor</Loc><Time>2010-05-13T11:53:08.008Z</Time>
<Device>R10001</Device><Zone>ShipZone</Zone><Org>Home</Org><Tag><ID>this is Auto print handler testing document</ID><Antenna>0</Antenna>  <TagReadTime>2010-05-13T11:53:08.008Z</TagReadTime><Accuracy></Accuracy></Tag></event_data>";

printRequestVO.setXmlDoc(new String[]{docToPrint}); // set Data to be printed

printRequestVO.setPrinterName("HP Universal Printing PCL 6"); //Set Printer name on which data to
// be printed
printRequestVO.setPrinterLoc("Hyderabad"); // Location of the printer

printRequestVO.setPrintDocName("Doc1"); // Print Document Format to be used to  print
printRequestVO.setCustomer(null); // Set the name of the customer
printRequestVO.setMaterialType(null); // Set the Material Type.
printRequestVO.setMaterial(null); // Set the material number.
printRequestVO.setSite(null); //set the location.
```

****

- Create the com.insync.edgeware.sensornet.printer. DocumentPrinter object and call the "printDocument" method of above created object by passing the, "printRequestVO" object

****

```
insync.edgeware.sensornet.printer.DocumentPrinter documentPrinter = new insync.edgeware.sensornet.printer. DocumentPrinter();
documentPrinter.printDocument(printRequestVO);
```

****

The above method will print the document at the location "Hyderabad" and on printer "Universal Printing PCL 6"

- **Example to print text with xslt template**
  - If the text to be printed is "This is an example for the print server"
  - The code looks like

****

```
insync.edgeware.sensornet.printer.DocumentPrinter documentPrinter = new insync.edgeware.sensornet.printer. DocumentPrinter();

String text = "This is an example for the print server";
```

String xslt = null; // Read particular xslt file and convert it as string form.
byte[] printData = documentPrinter. getPrintText(text, xslt);
****

- The above method returns printData as byte which is of pdf type, and can be printed using third party libraries which prints PDF type of files.

- **Example to print text without xslt**
  - In this case the xml should be of the below format

    <?xml version='1.0' ?><PrintData><Data>text</Data></PrintData>

  - The code is looks like

****

    insync.edgeware.sensornet.printer.DocumentPrinter documentPrinter = new insync.edgeware.sensornet.printer. DocumentPrinter();

    String text = "<?xml version='1.0' ?><PrintData><Data> This is sample text to be printer'</Data></PrintData>";
    byte[] printData = documentPrinter. getPrintText(text, null);

****

  - The above method returns printData as byte which is of pdf type, and can be printed using third party libraries which prints PDF type of files.

- **Example to print cml data with xslt**
  - In this case care should be taken to match the xml to be printed with the xsl file that is used to do the transformation.
  - The code is looks like

****

    insync.edgeware.sensornet.printer.DocumentPrinter documentPrinter = new insync.edgeware.sensornet.printer. DocumentPrinter();

    String text = null; // XML should be according to xslt, that parses it.
    String xslt = null; // Read particular xslt file and convert it as string form. And it //should be able to conver given xml.

    byte[] printData = documentPrinter. getPrintText(text, xslt);

****

  - The above method returns printData as byte which is of pdf type, and can be printed using third party libraries which prints PDF type of files.

4.13 Event Management Service

Event Management Service (EMS) in Sensornet manages the set up of Events, Subscribers, Subscriptions and Jobs at each Enterprise Organization from a User Interface stand point.

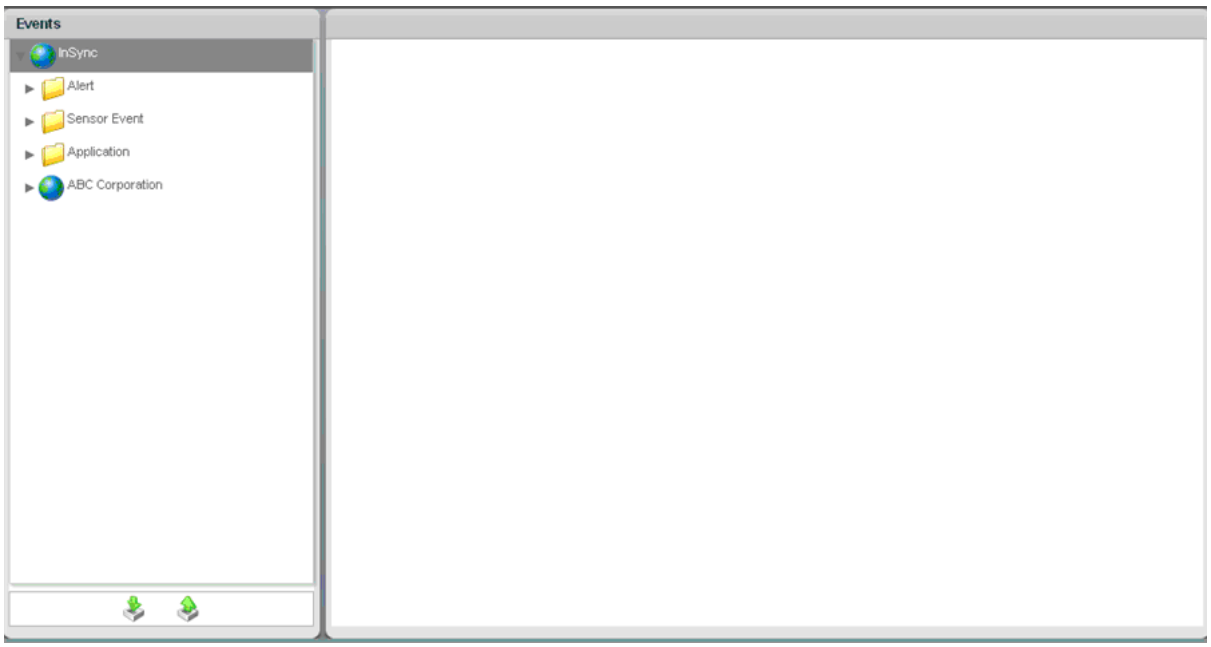
At a different level altogether EMS works seamlessly on the server side in processing the RFID sensor data coming in from the devices configured in Configuration (explained in section 4) according to the Event Model configured in the EMS UI. Based on the Event Model appropriate events are raised by EMS. The Event Model includes the events, the handlers, the actions that need to be taken, subscriptions and the jobs that are setup for the event.

Note: Events without a subscription or a job or a data handler will not get raised. An event created should at least have one of the above objects defined on the event for the system to raise it to be used in the process.

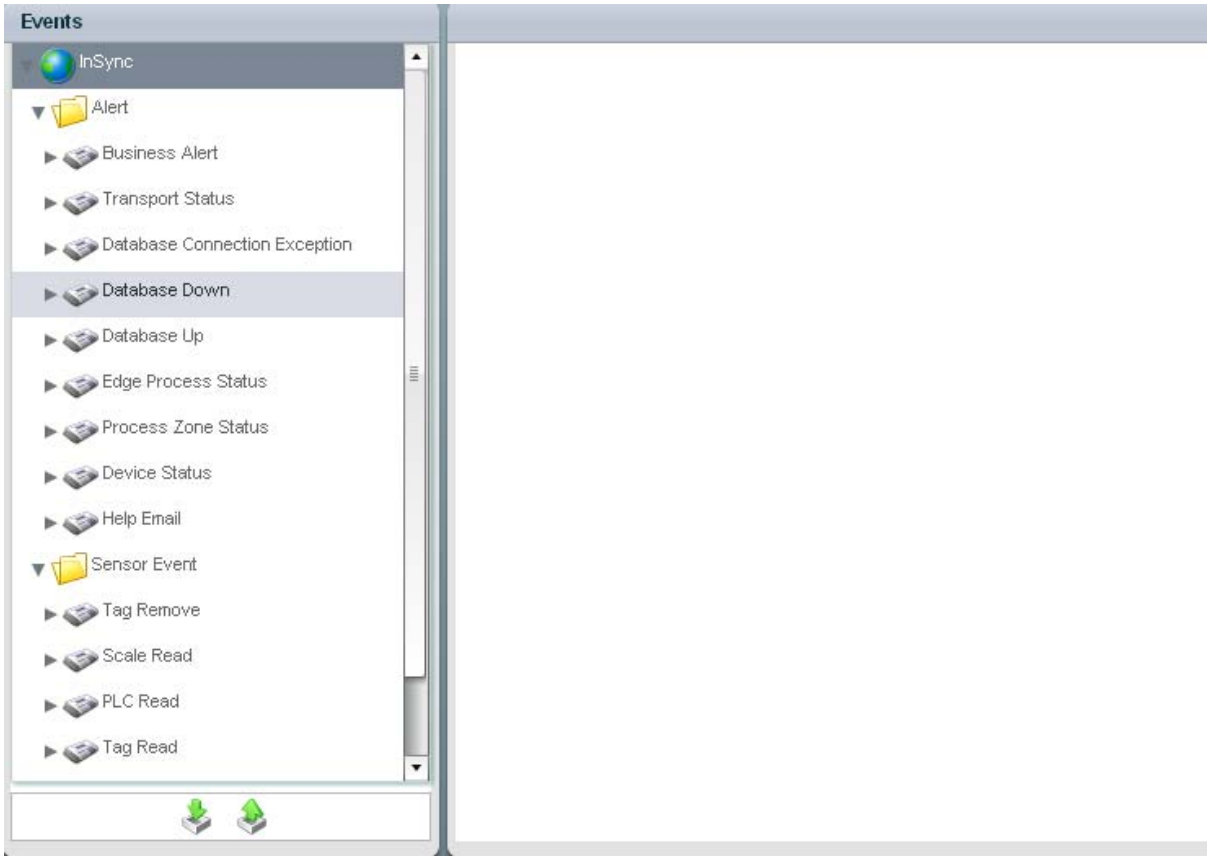
Click on in the first level menu.

4.13.1 Events

An Event is a message in iApp which consists of an event name and associated data with that instance.



- Events in iApp are of types - Alerts, Sensor Events and Application Events.
- iApp provides predefined Alerts and Sensor Events that can be used across all Enterprise Organizations.

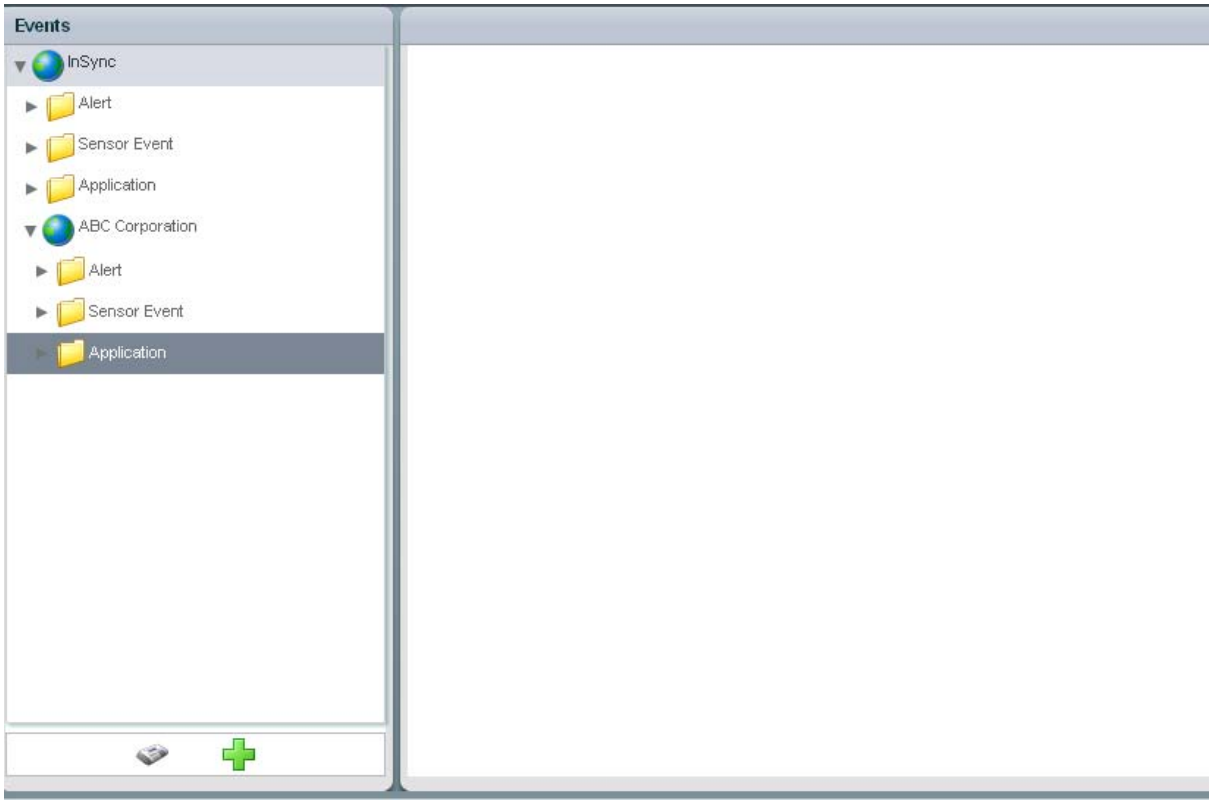


- Application Events are specific to the Enterprise Organization that they are created for.
- Only Administrative user at the Home Enterprise will be able to create / modify the Alert Events and Sensor Events
- Application Events can be created by the Users belonging to that Enterprise organization
- Each Event can be configured with a set of Actions and Subscriptions.

4.13.1.1 Create Event

To create a new Event

- Select the Enterprise Organization, select the type of Event and click on
- For Application type Events multiple application names (domains) can be setup by clicking on and events created for each of the domains. Note that the Domains can be created up to 2 levels only.
- By default iApp provides a ‘Pack Complete’, Periodic Plan Expired Event and Periodic Plan Warning Event that can be configured.

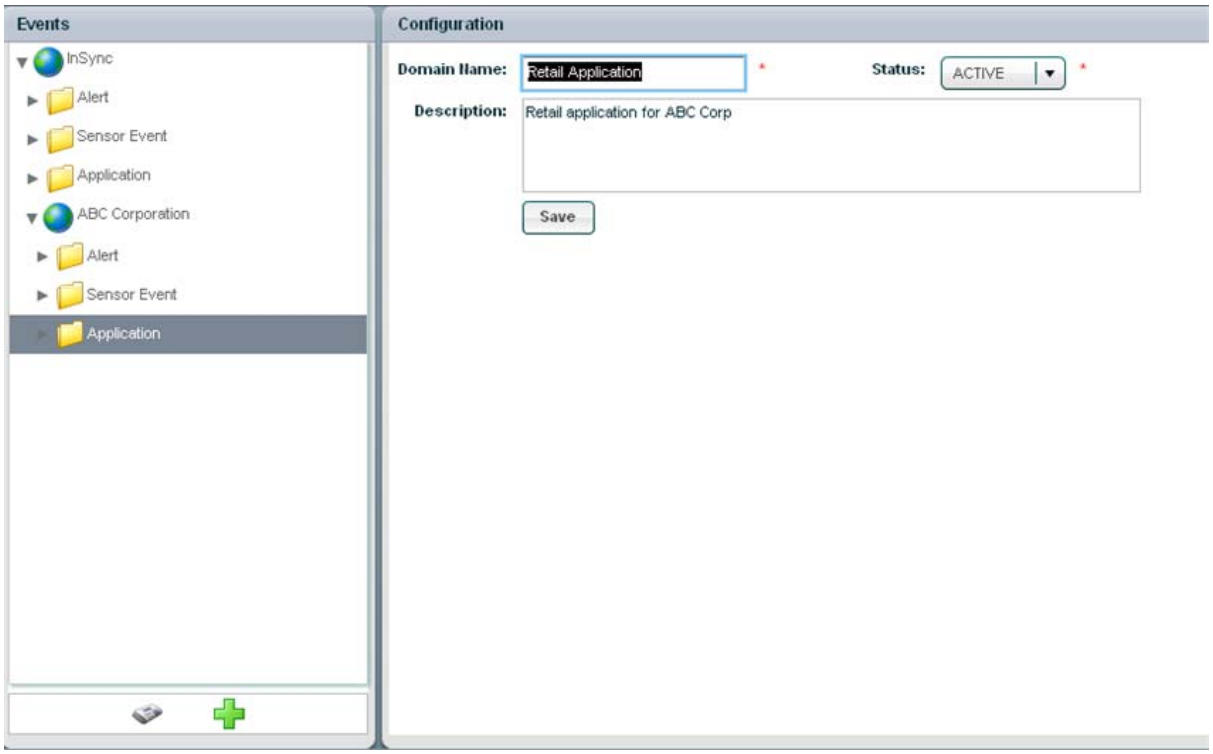


Operations possible on the Application are

- Create Event
- Create Domain

For purpose of documentation we will create an application Event within a domain.

Select Application and click on .



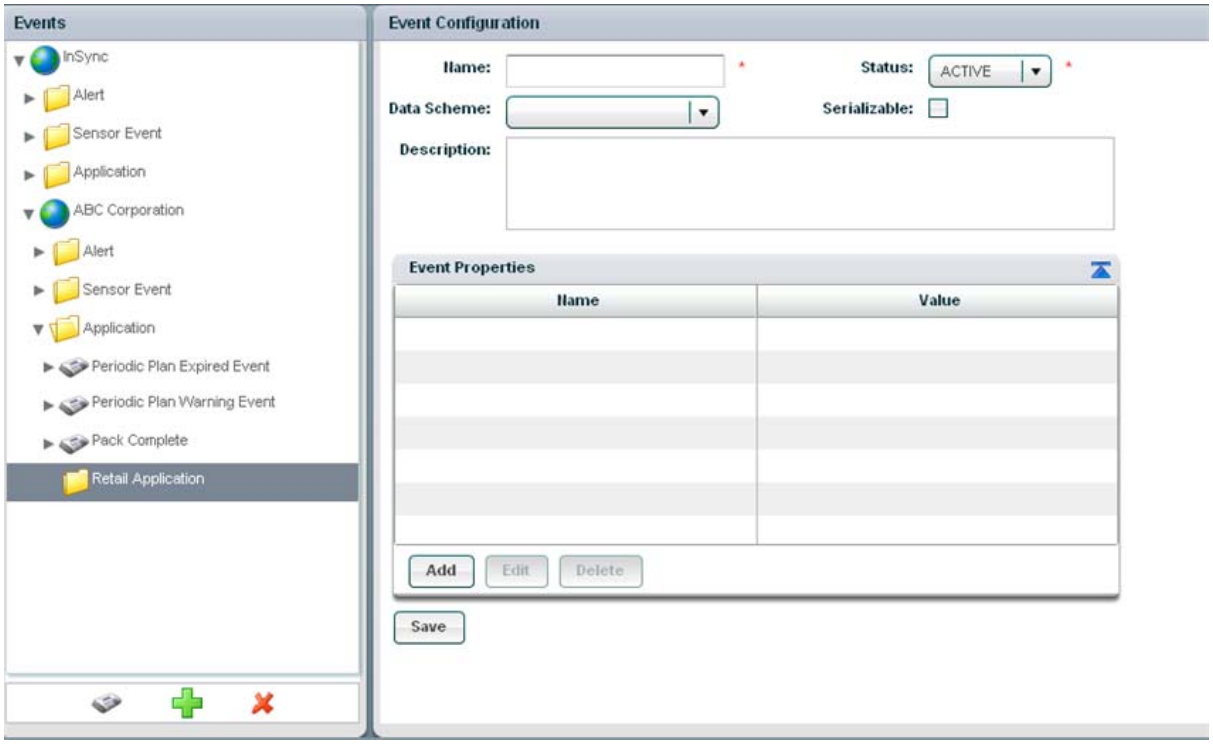
The different attributes for the Domain are:

Attribute	Description
Domain Name	Unique name for the domain within the Enterprise
Status	Active or InActive
Description	Short description

Note :

1. 1. The system allows only 2 levels of Application Creation
2. 2. The Applications created here are linked to the Catalogs which will be explained in detail in the Administration Section. The Applications can also be created using the Catalogs

To Create an Event Select the domain that was created and click .



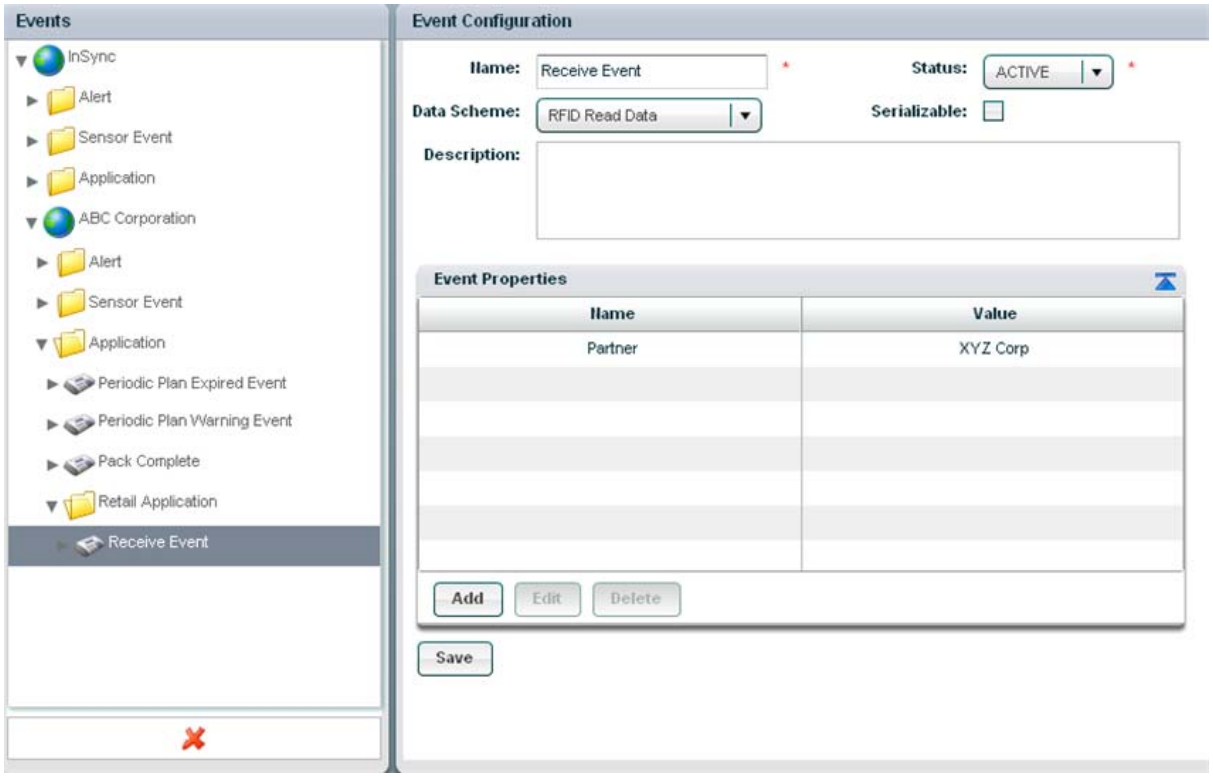
The different attributes for the Event are:

Attribute	Description
Name	Unique name for the Event within the Enterprise Note that the event name should not be created with any special characters.
Status	Active or InActive. If set to InActive the event is not processed. On a running system changing the status from InActive to Active will result in only the new events getting processed.
Data Scheme	This is the schema responsible for converting the sensor data (which is in the form of name value pairs) into a particular data format. It is an XML schema which defines the data elements and the structure. It is not mandatory to have a data schema. There might be instances where the data itself is already formatted. Data Schemes are created using the Data Schemes tab.
Serializable	Check if serialization of events is required
Description	Short description
Event Properties	These are properties in the form of name value pairs that can be added at runtime into the event data. Properties that are added here should be added

	as elements to the data schema. The values are static values and more than one value can be set for a property.

Click ***Save*** at the bottom of the page to save the event.

Note: Avoid using special characters for the event name



In the above example Partner property added to the Event Properties should be part of the RFID Read Data Schema which otherwise do not contain the property.

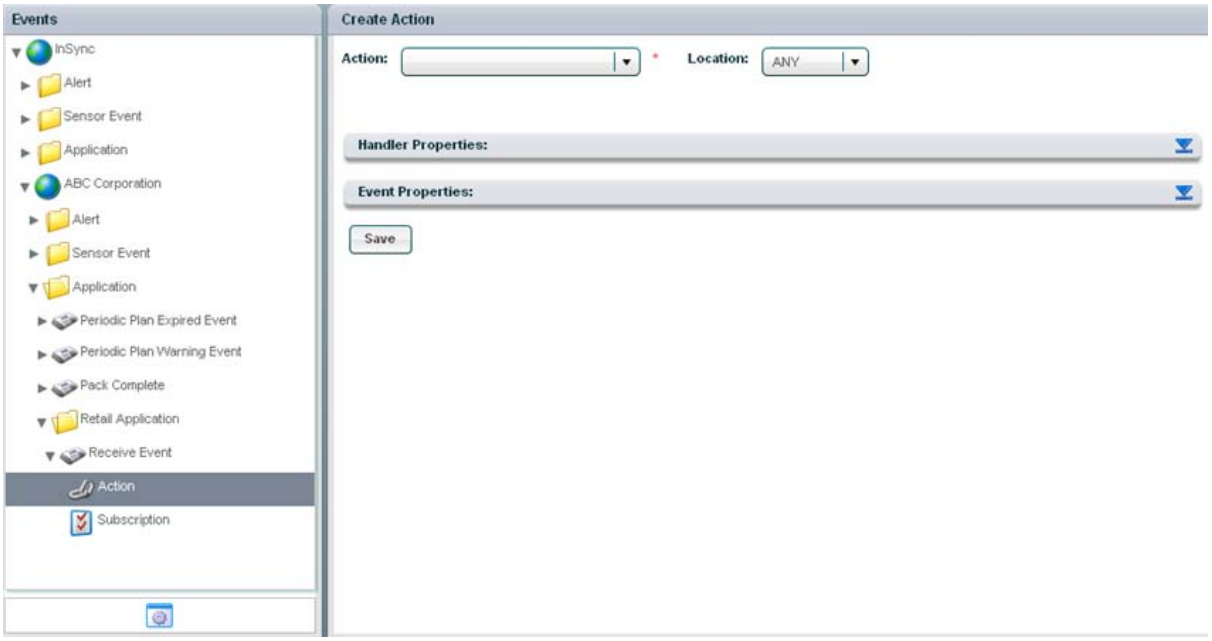
- Click to minimize or maximize the panel
- To edit or delete the Event property, select the property and click on the appropriate button.

4.13.1.2 Create Actions

Actions are set of activities that occur on the generated event. One or more actions can be set for the event and the order of execution of the actions can be controlled.

Select and click on the event where the action needs to be defined. On click the event tree expands.

Select Action and click on to create an action.

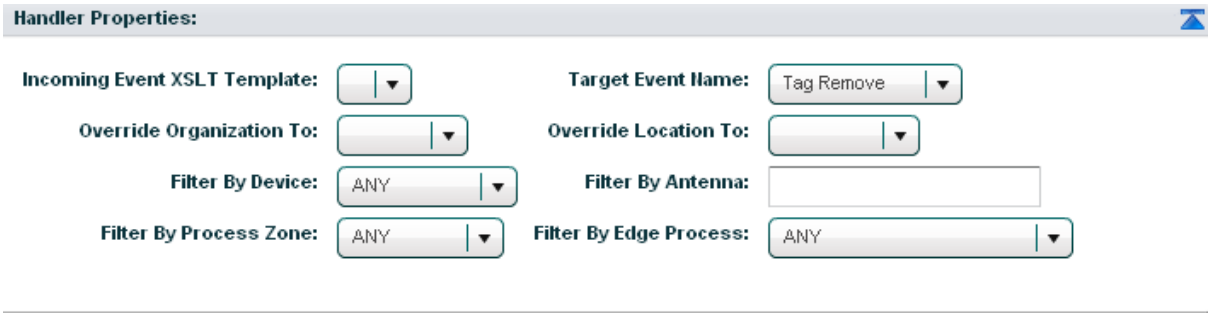


Attribute	Description
Action	is the data handler that will get executed when the event gets raised
Location	is the location from where the event gets raised. Only events coming in from the location will be acted upon. ANY location will mean all events from any of the locations.

Handler Properties

These are the properties for which the values can be provided to the handler at run time. The data coming in from the event can be over ridden by the property values set.

The Handler properties vary from handler to handler. Each handler comes in with a setup of properties for which values can be set.



For more information on the handler properties refer to IApp Event Handlers section.

Event Properties

These are the properties of which the event is made of.

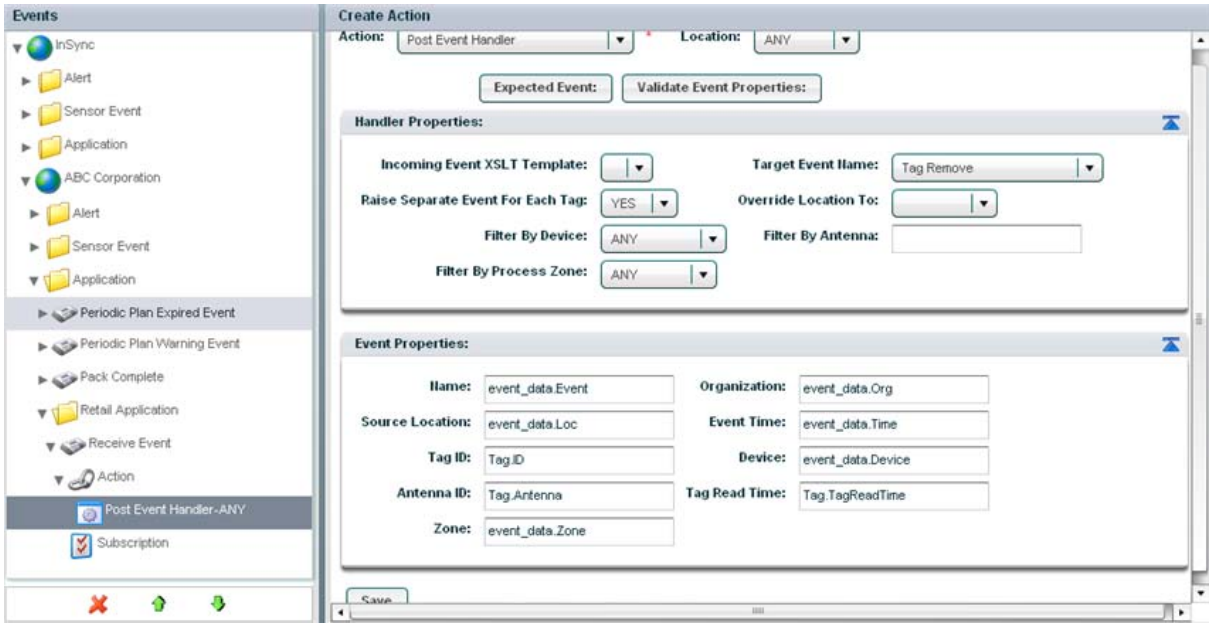
The values for each of the Property Names are the element names in the Event which gets acted upon by the Handler configured.

Note: It is recommended that the Event properties are not modified while creating the actions.

Note:

1. Refer to iApp Event Handlers section for more information on Event Handlers.
2. Refer to iApp Event Handler Design section to create a new event handler

Click on ***Save*** to save the Action. The Action gets saved as <handlername> - <LocName> in the Event tree.



- More than one action can be set on the Event
- The order of the execution of the Actions can be set by using the and buttons in the left side lower panel.
- On Save of the Action two buttons get enabled on the Action page.
    - Expected Event

- Validate Event Properties

The Expected Event shows the event XML of the event for which the Action has been created. This is basically the Data Schema format configured to be used for the Event. The purpose of the button is purely for verification purposes only.

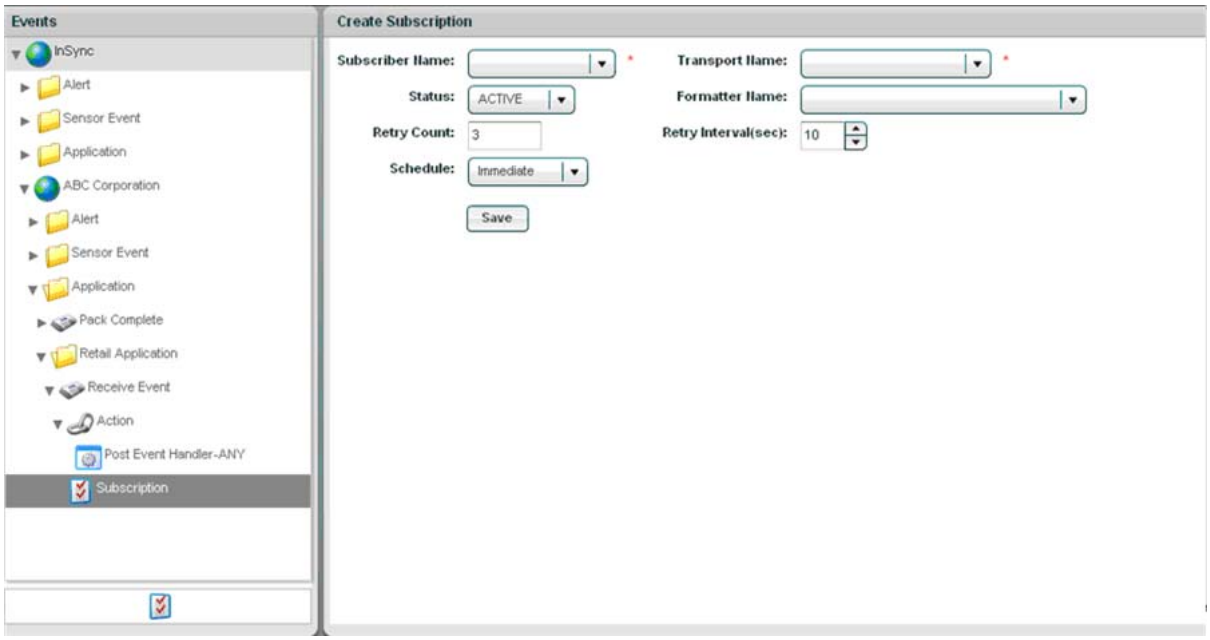
The Validate Event Properties validates the Event Properties that are set for the Action. This is to make sure that the Event properties if modified are according to the expected Event elements shown in the Expected Event pop up. The purpose of the button is for verification purpose only so that based on the validation necessary corrections can be made.

The body of the message gives an indication of whether all the properties are matching to the data schema or not. If there is a property that is missing which is part of the data schema then the appropriate message will be shown.

4.13.1.3 Create Subscriptions

Subscriptions in IApp refer to the process of subscribers signing up for Events generated in the system. Subscribers and the mode of transport for the Subscription should be defined before the Subscription can be created.

Select Subscription for the Event and click on .



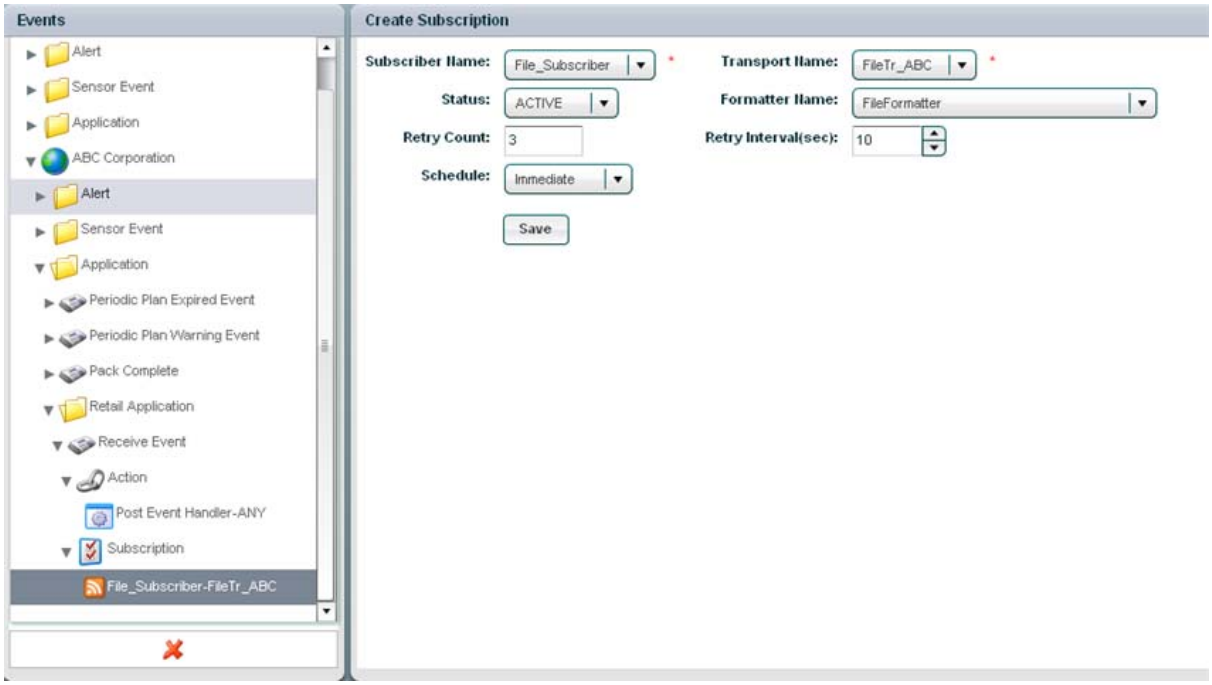
The attributes for the Subscription are:

Attribute	Description
Subscriber Name	All the subscribers created from the Subscribers tab are listed here.
Transport Name	These are Transports for a given subscriber that would be used while delivering the events to the subscribers. Transports are created while creating the Subscribers from the Subscriber tab.
Status	Active / InActive If the status is set to ‘InActive’ for a schedule set to ‘Immediate’ then the subscription is not processed. If the status comes ‘Active’ again then all subscriptions to the events raised from that point onwards will get processed. If the status is set to ‘InActive’ for a schedule set to ‘Recurrence’ then the subscription is not processed for the duration it is set to InActive. If the status comes ‘Active’ again then the subscription gets processed for all the events raised during the InActive period also. To disable the Subscription totally you will have to delete the subscription.
Formatter Name	Formatter that should be used for the subscription. Formatters are templates that are defined to format the events to be delivered to the subscribers and will contain element variables which are sourced from the Event Schema. Formatters are created from the Formatters tab.
Retry Count	No of times the system will try to deliver the event to the subscriber

	if it fails the first time around.
Retry Interval	Frequency of each retry.
Schedule	Can be immediate / schedule. Based on the schedule the events will be delivered to the subscribers. Immediate - events will be delivered as and when the events are generated. Recurrence – scheduler based subscription where in the events will be delivered to subscribers at the scheduled time and date.

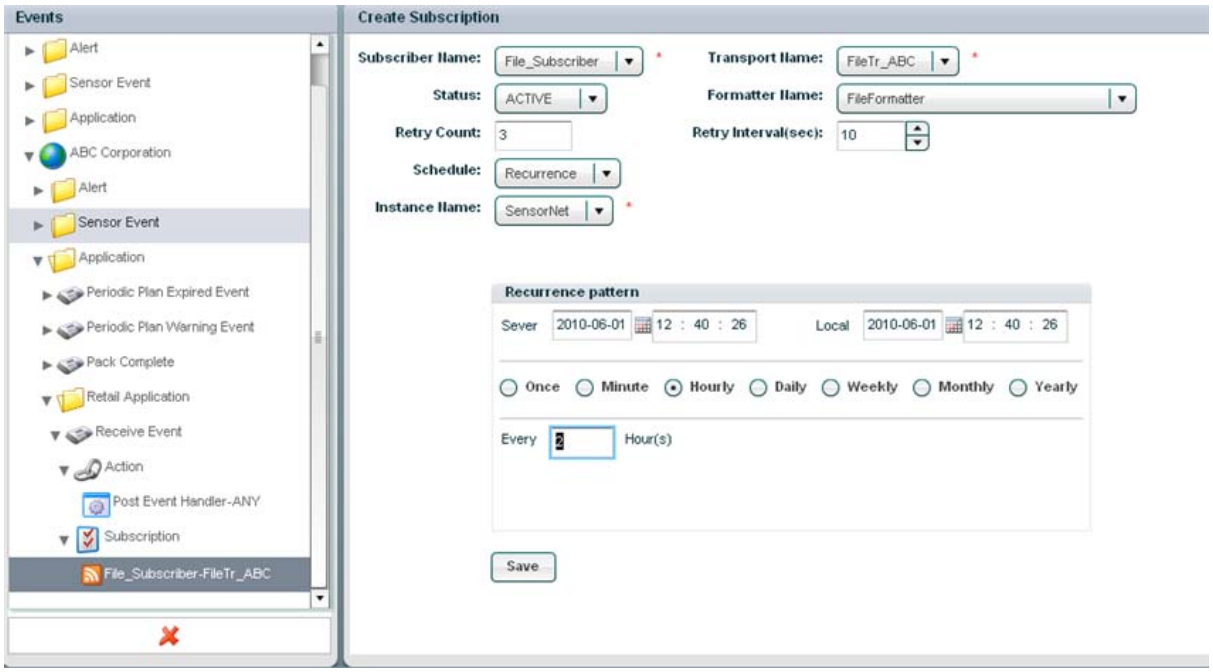
On click of ***Save*** the Subscription is saved and is added to the left side event tree as <SubscriberName>-<TransportName>

Immediate Subscription:



Based on the above subscription events get delivered immediately as and when they are raised. The events raised are formatted using the FileFormatter and the formatted event is delivered to the subscriber using the File Transport. If the subscription fails to deliver it will retry at an interval of 10 seconds for 3 times and gives up if it fails to deliver.

Recurrence Subscription:

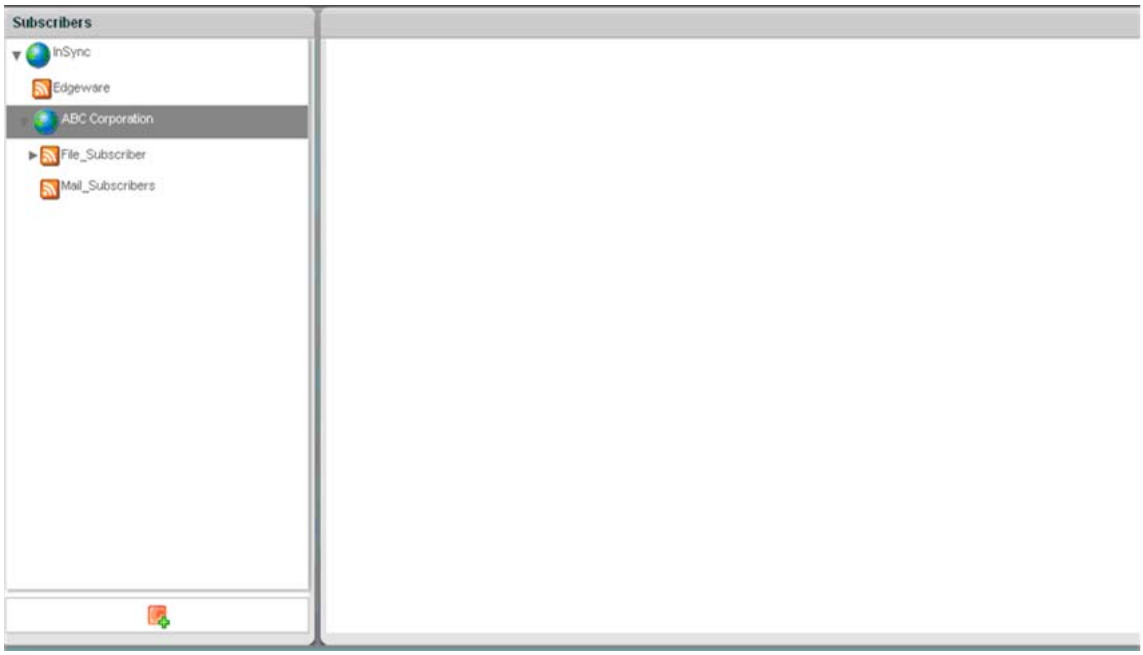


- Based on the above subscription events get delivered to subscribers every 2 hours from the start date.
- Note the Server and the Local times shown in the Recurrence Pattern panel. The schedule is run against the server time. The Local time is shown so that configuring the schedule would be easier for the user.
- The instance name drop down lists all the sensornet instances registered. The scheduler runs against the sensornet instance selected from the drop down. This becomes important when there are 2 primary sensornet instances registered.

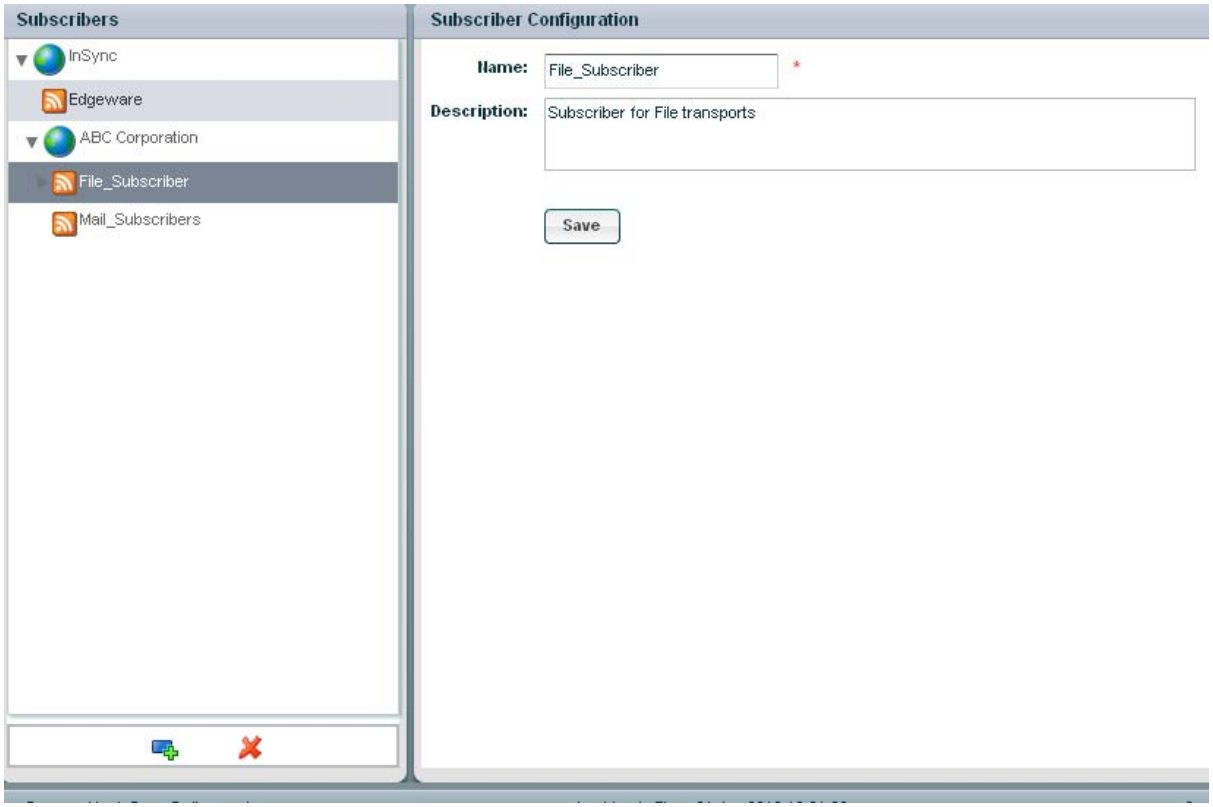
4.14 Subscribers

Subscribers are entities interested in subscribing to events in IApp. Each Subscriber can have one or more transports created that will be used during the delivery process.

Subscribers in IApp are created by Enterprise Organization.



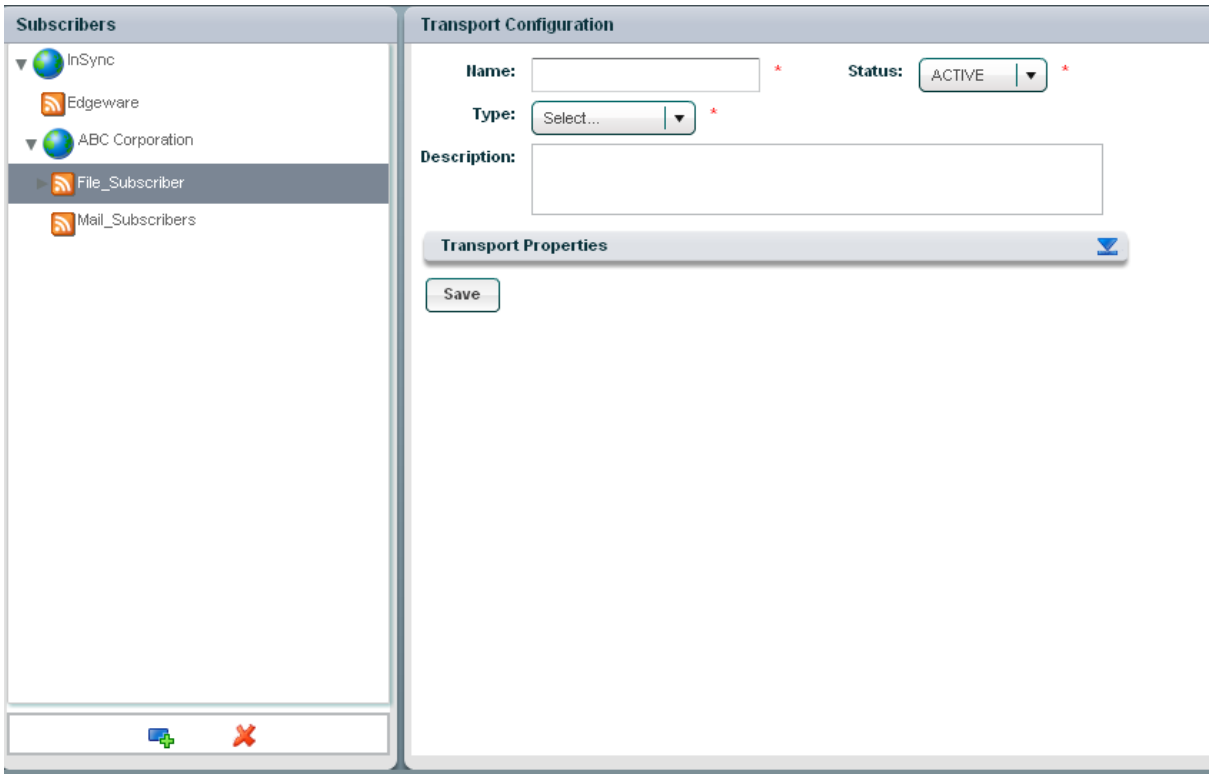
Select the Enterprise Organization for which the Subscriber has to be created and click



Operations possible on the Subscriber are

- Create Transport
- Delete Subscriber

To create a **Transport** click on . Each Subscriber can have more than one transport created with different transport types.

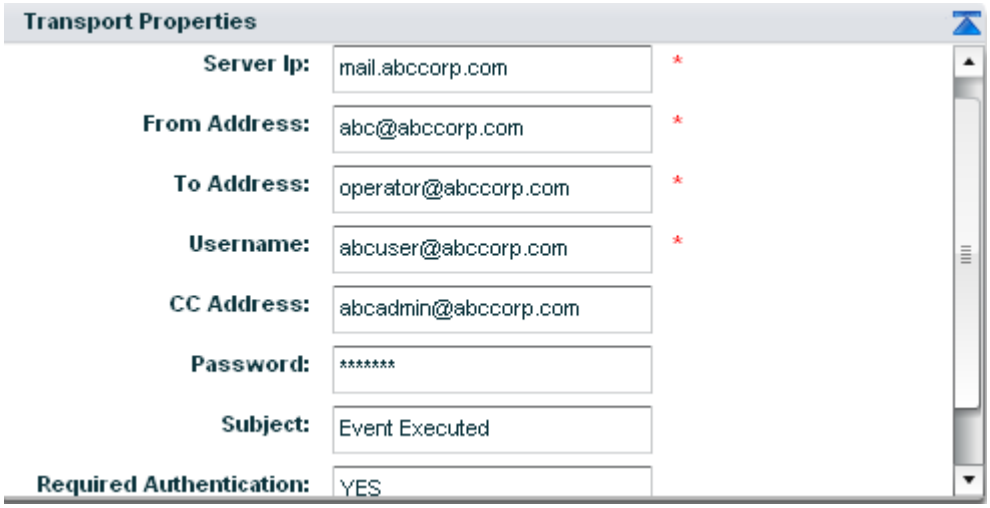


The attributes for the Transport are

Attribute	Description
Name	Unique name for the Transport
Status	Active / InActive Same as Subscription 'Status'.
Type	Type is the transport type to use while transporting the event to the subscriber. The transport properties change based on the type selected from the Type dropdown. The different transport types available are: Email / FTP / File / HTTP / HTTPS / IDOCTransport / JMS / MSMQ / SAP Java / SOAP / TCP / Yahoo Messenger
Description	Short Description
Transport Properties	Are the different properties captured based on the type selected. The properties panel gets displayed with the appropriate properties based on the type selected.

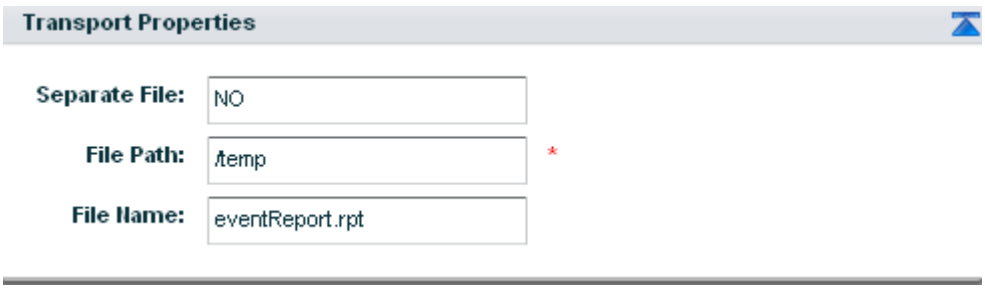
Transport Properties: The transport properties change from type to type. Based on the type selected in the drop down the.

For Type **Email** the properties captured are



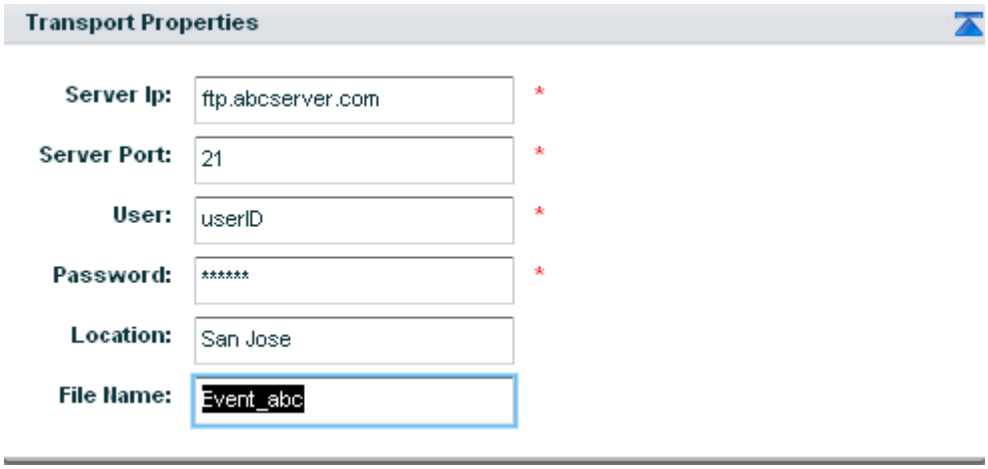
Attribute	Description
Server Ip	IP address of the mail SMTP server
Server Port	Port of the smtp server
Required Authentication	Yes / No. Yes if the mail server requires authentication
Username	Mailing User name of the user on the mail server for authentication
Password	Password for the username
Mail separator	
CC Address	Additional Copy mailing address list
Subject	Subject to be added in the email
From Address	the senders email address for the mail
To Address	Mail recipient addresses. Multiple mail addresses can be added by a comma separator
CC Address	Mail copy recipient addresses. Multiple mail addresses can be added by a comma separator
Subject	Subject of the mail
Content Type	Represents how the data will be encapsulated.

For Type **File** the properties captured are:



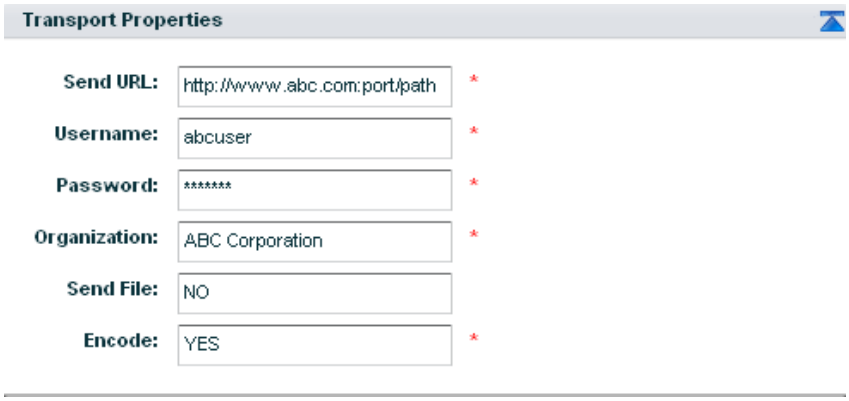
Attribute	Description
Separate File	YES / NO based on whether the events have to be written to a single file or separate file every time
File Path	Path where the file has to be saved. Note: The path represented here is not the local path but the path on the server where Sensornet is running.
File Name	Name for the file

For Type **FTP** the properties captured are



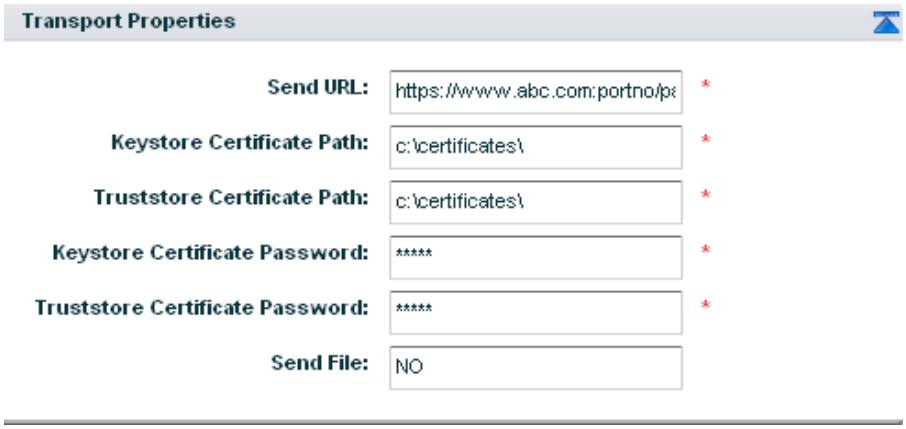
Attribute	Description
Server Ip	IP address of the FTP server
Server Port	Port No of the ftp server
User	Valid User name on the FTP server
Password	Password for the user
Location	Location of the FTP server
File Name	File to be transported

For Type **HTTP** the properties captured are



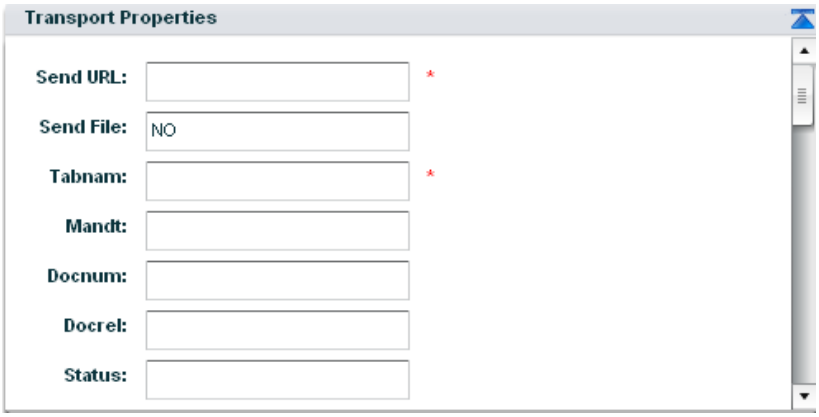
Attribute	Description
Send URL	http URL to post
User name	Username for authentication
Password	Password of the user
Organization	Enterprise organization
Send File	Yes / No
Encode	Yes / No – Yes to encode the event contents being posted

For Type **HTTPS** the properties captured are



Attribute	Description
Send URL	https URL to post
Keystore Certificate Path	Path of the certificate on the file system
Truststore Certificate Path	Path of the certificate on the file system
Keystore Certificate Password	Password for the keystore
Truststore Certificate password	Password for the truststore
Send File	Yes / No – Yes to send the event contents as a file

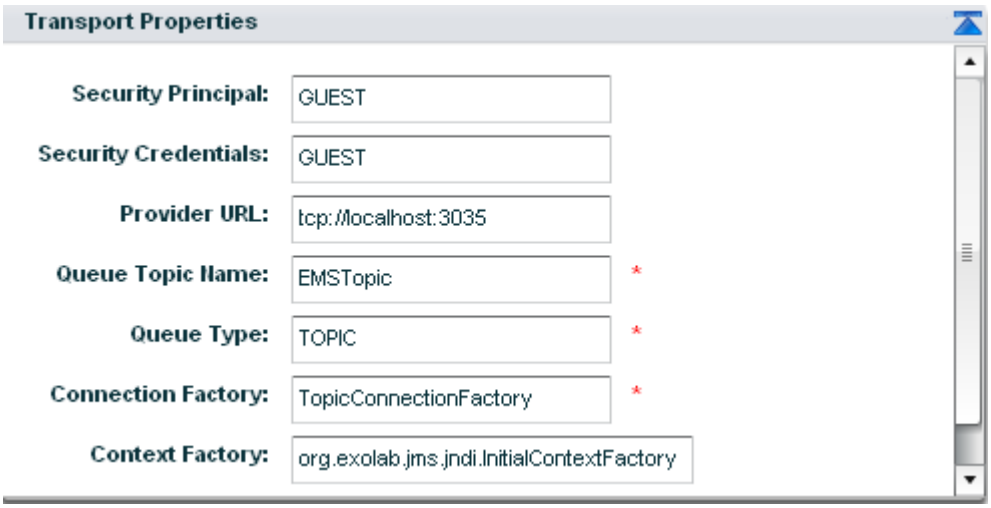
For Type **IDOCTransport** the properties captured are



Attribute	Description
Send URL	URL for the service in SAP WAS which accepts XML IDOCS
Send File	Yes / No
Tabnam	Control Segment Structure. This is fixed by IDOC release version. Example - EDI_DC40
Mandt	SAP client the IDOC is directed to. The value can be 100
Docnum	IDOC Number - Generated in SAP as IDoc saved to database. Can be useful to generate your own DOCNUM to link together the Control Segment to associated Data Segments for custom processing outside of SAP.
Docrel	SAP Application version
Status	Direction of IDOC generated in SAP
Direct	Direction of IDOC. 1 meaning OutBound and 2 meaning InBound
Outmod	Output mode used by SAP
Exprss	Inbound processing override flag
Test	Test Flag - If the test flag is set in the IDoc, it must also be set in the Partner Profile.
Idoctyp	IDOC basic type - Basic type for Payment Advice 820.
Cimtyp	Custom Extension – Used if custom extension built for IDoc Basic Type in SAP.
Mestyp	Logical Message Type - Logical Message Type for Payment Advice 820. This field is part of the key used to identify the Partner Profile in EDP21.
Mescod	Message Code - When populated, MESCOD becomes part of the key used to identify the Partner Profile in EDP21. It is generally used when custom processing in SAP requires additional Partner Profiles for a particular Partner-Logical Message combination.
Mesfct	Message Function - When populated, MESFCT becomes part of the key used to identify the Partner Profile in EDP21. It is generally used when custom processing in SAP requires additional Partner Profiles for a particular Partner-Logical Message combination.
Std	EDI Standard - Optional. X = X12, E = EDIFACT, etc
Stdvrs	Edi version - EDI Transaction or Message Type version
Stdmes	EDI message type - EDI Transaction or Message Type. Useful for identifying IDocs and Envelopes in the EDI subsystem and to support EDI reporting in SAP
Sndpor	Sender Port - File Port Defined in WE21 for inbound as Sender and outbound as Receiver. Checked by EDI_DATA_INCOMING before IDoc file processed
Sndprt	Partner type of sender - References the Customer Partner Type. This field is part of the key used to identify the Partner Profile in EDP21
Sndpfc	Partner function of sender - When used, SNDPFC becomes part of the key that identifies the Partner Profile in EDP21.
Sndprn	Partner number of sender - SAP Customer. Must exist in the Customer Master. SNDPRN is part of the key used to identify the Partner Profile in EDP21
Sndsad	Sender Address - Don't populate: reserved for future use by SAP.
Sndlad	Sender logical address - EDI Trading Partner ID of Sender (Customer). Very useful as a way to tighten the link between the IDoc and the EDI Transmission.
Rcvpor	Receiver port - Anything can be inserted here but consistency is highly recommended. Convention is to use SAP

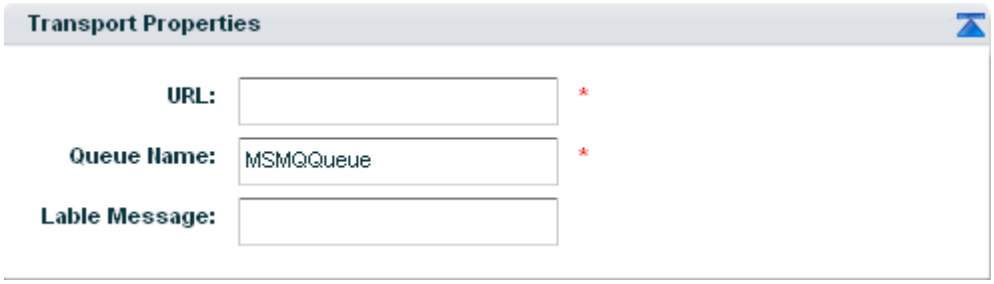
Rcvprt	Partner type of recipient - Logical System. Defines Receiver Partner on the inbound
Rcvpfc	Partner function of recipient - Don't populate.
Rcvprn	Partner number of recipient- SAP Logical System for receiving client
Rcvsad	Recipient address (SADR) - Don't populate: reserved for future use by SAP.
Rcvlad	Recipient logical address - EDI Trading Partner ID of Receiver (us).
Credat	Created Date- IDoc create date. SAP generates this when the IDoc is saved on the IDoc database at status 50. Anything mapped here will be overwritten by SAP.
Cretim	Time created - IDoc create time. SAP generates this when the IDoc is saved on the IDoc database at status 50. Anything mapped here will be overwritten by SAP.
Refint	EDI Interchange Control Number – It creates an explicit link between the IDoc and the EDI transmission. This number eases Production Support greatly. When there's an issue with a Partner EDI transmission, the Partner's EDI support people want to know the Interchange Control Number.
Refgrp	EDI Group Control Number - Identifies the EDI group. If the ISA Control is included, this is not really necessary although you can bring it in and it does identify the Group within the Transmission that was mapped to the IDoc.
Refmes	EDI Transaction Control Number - Identifies the EDI transaction within the Group. I generally use this to identify other information about processing within the EDI subsystem such as GIS BP ID.
Arckey	Key for external message archiving. Can be used to store additional information about processing in the EDI subsystem.
Serial	Serialization field - Used by SAP to determine order IDocs should be processed when imported as a batch referencing a common Partner Profile.

For Type **JMS** the properties captured are



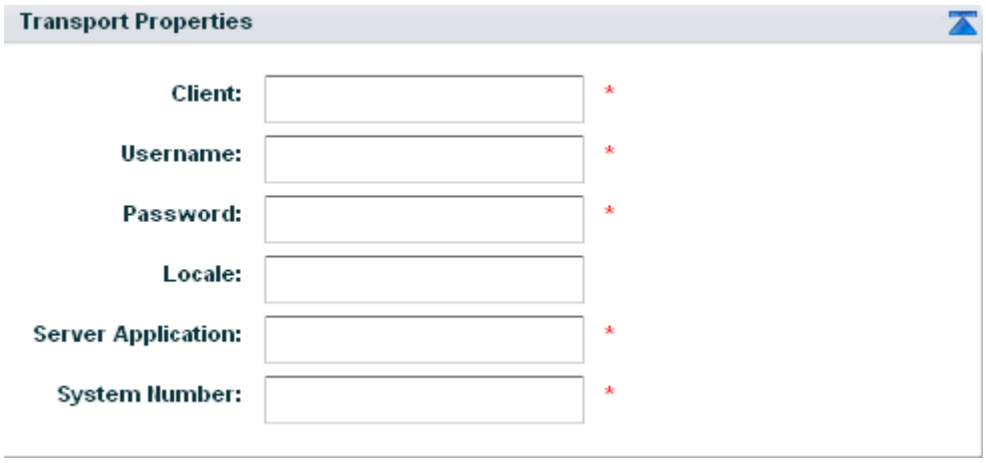
Attribute	Description
Security Principal	Jndi user
Security Credentials	Password for the jndi user
Provide URL	url where JMS is running
Queue Topic Name	Queue / Topic name
Queue Type	Type of the queue Topic / Queue
Connection Factory	JMS connection fatctory
Context Factory	JNDI context factory
Required Security	Yes / No

For Type **MSMQ** the properties captured are



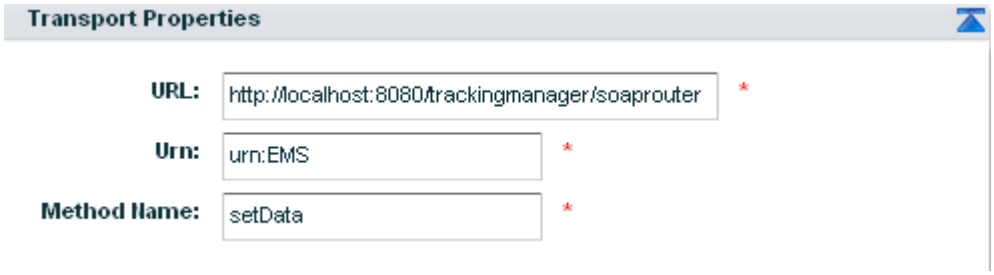
Attribute	Description
URL	Location of the Queue
Queue Name	Queue name
Label Message	Describes the type of service provided by the Queue

For Type **SAPJAVA** the properties captured are



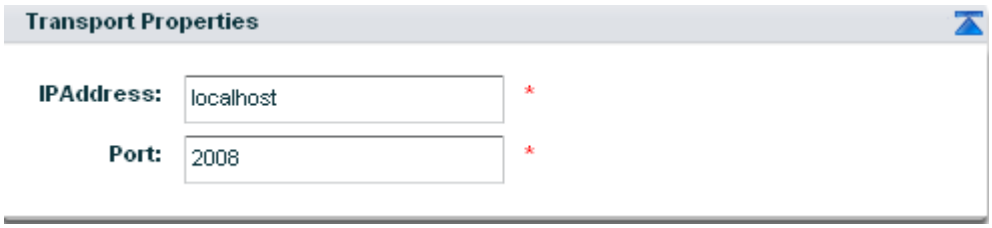
Attribute	Description
Client	SAP client location
Username	Valid username on the SAP Server
Password	Password for the user
Locale	Language setting
Server Application	IP or fully qualified name of the SAP server
System Number	The two digit system number for the SAP installation

For Type **SOAP** the properties captured are



Attribute	Description
URL	Location of the soap client where the data has to be sent
Urn	Unique name for the service running on the SOAP server.
Method Name	Method to be executed on the soap client

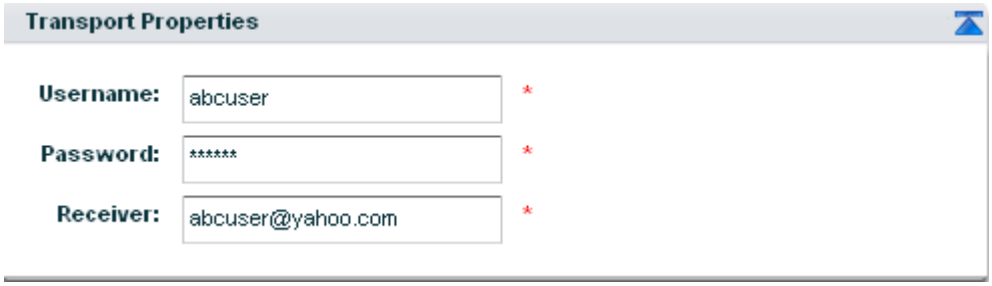
For Type **TCP** the properties captured are



Attribute	Description
IP Address	Address of the TCP client
Port	Port where the data will be posted on the IP Address

For

Type **YahooMessenger** the properties captured are



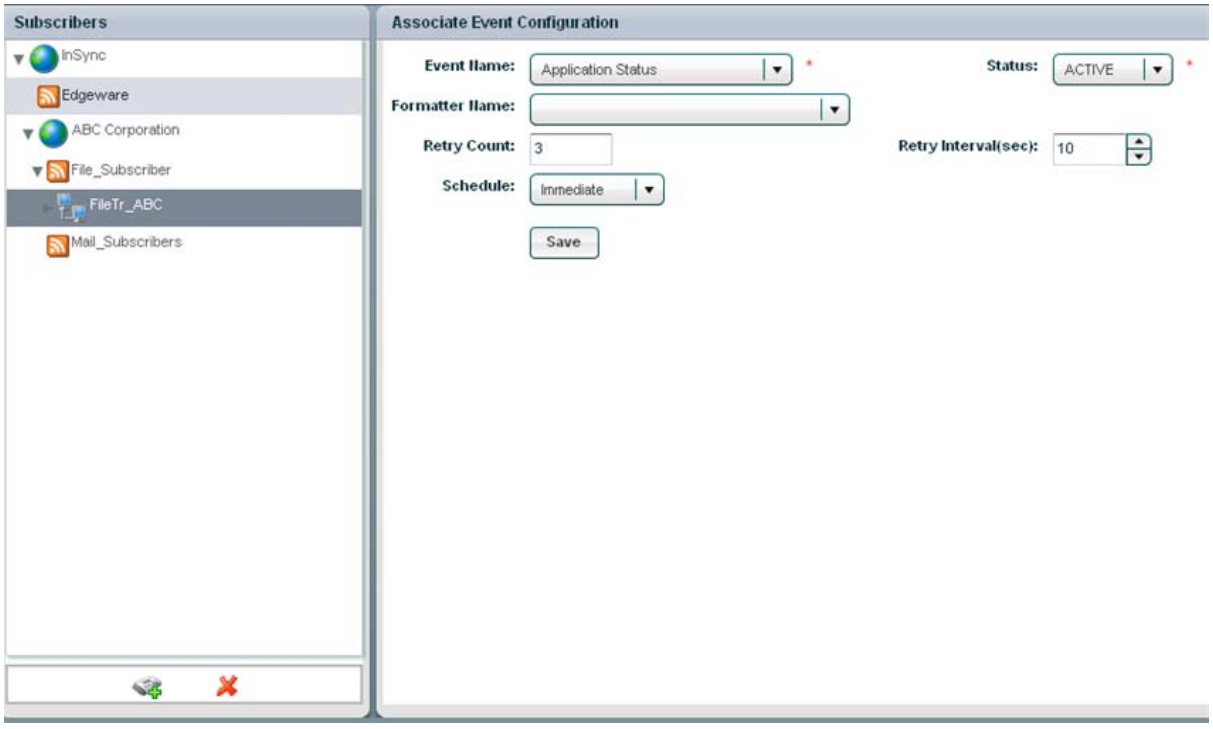
Attribute	Description
Username	Username of the yahoo messenger user
Password	Password for the user
Receiver	Receiver yahoo messenger id

Click ***Save*** to save the transport. The transport gets added to the Subscriber in the left side panel.

Operations possible on the transport are

- Create Subscription
- Delete Transport

To create a Subscription select the transport and click on .



Note: Subscriptions for an event can be created from the Events tab as discussed in the previous section. Association of the events to the subscribers is possible from this tab.

The attributes of the Subscription to a given transport are

Attribute	Description
Event Name	All Events created are listed in the drop down. Select the event to be subscribed
Status	Active / InActive
Formatter Name	Formatter that should be used for the subscription. Formatters are templates that are defined to format the events to be delivered to the subscribers and will contain element variables which are sourced from the Event Schema.
Retry Count	No of times the system will try to deliver the event to the subscriber if it fails the first time around.
Retry Interval	Frequency of each retry.
Schedule	Can be immediate / schedule. Based on the schedule the events will be delivered to the subscribers. Immediate - events will be delivered as and when the events are generated. Recurrence – scheduler based subscription where in the events will be delivered to subscribers at the scheduled time and date.

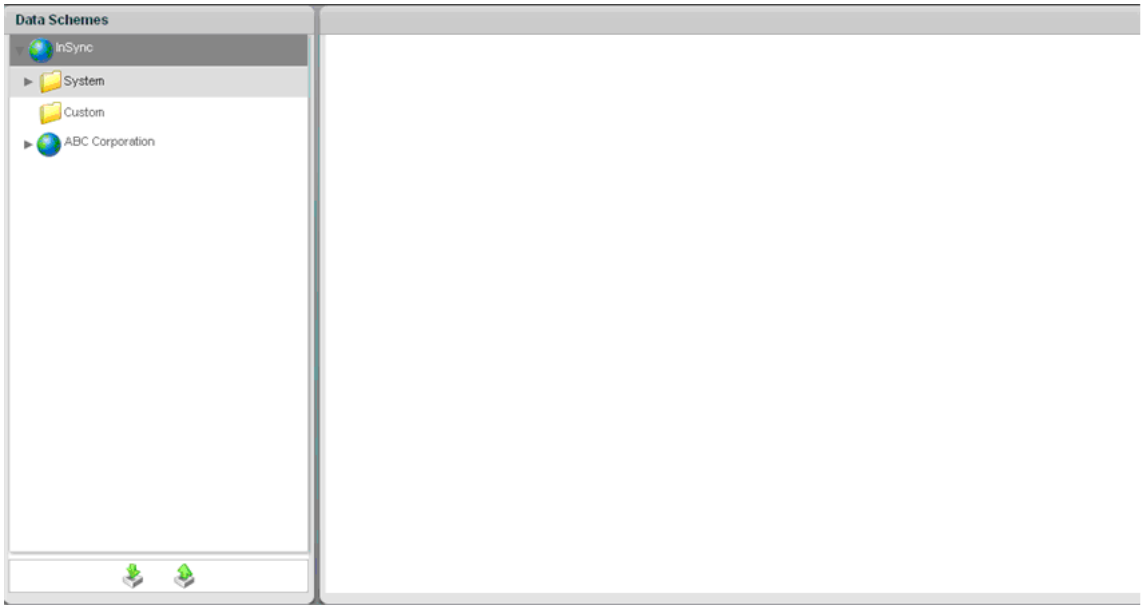
Note: For more information on the Schedule refer to **Create Subscription section**

4.15 Data Schemes

Data Schemes are responsible for converting the sensor data (which is in the form of name value pairs) into a particular data format.

It is an XML schema which defines the data elements and the structure. Each Event that will be raised in the system will have its own data structure represented by the Data Scheme that the event is configured with.

Data Schemes created are configured to an Event during the Event creation process from the Events tab.



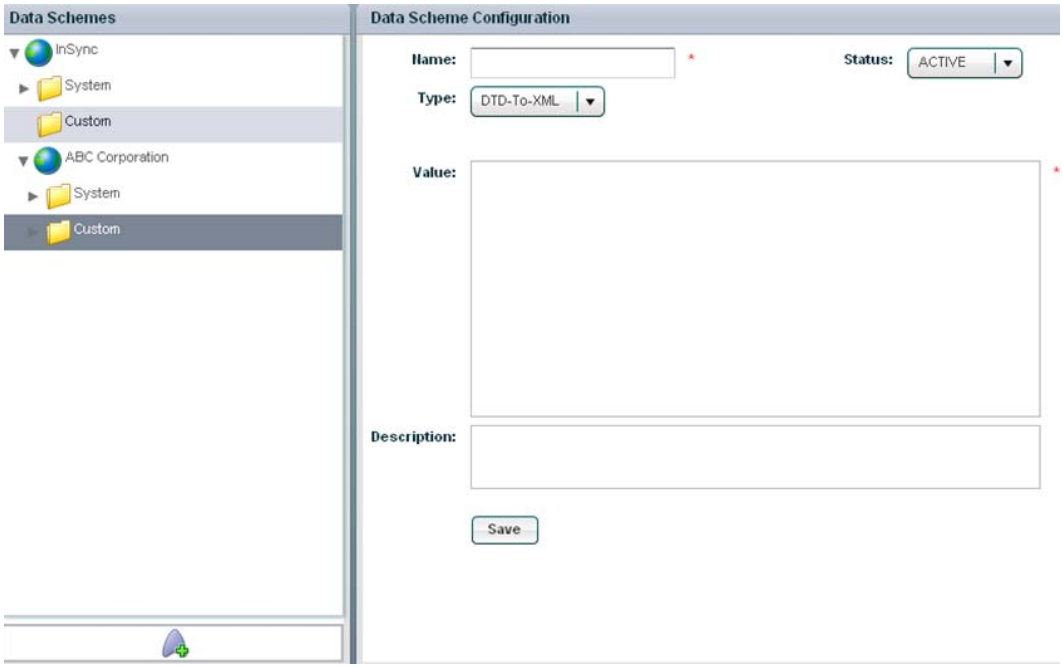
Data Schemes in IApp are listed by Enterprise – System / Custom.

- The System Data Schemes are provided by default and are available across all Enterprises.
- Only users with Administrative rights at the Home organization will be able to add or modify the System Data Schemes
- Users belonging to an Enterprise organization will be able to add new Custom Data schemes and modify the existing custom tag schemes in that Enterprise only.
- Administration users at Home Organization will be able to add custom Data schemes in other Enterprise organizations.

13.3.1 Create Data Scheme

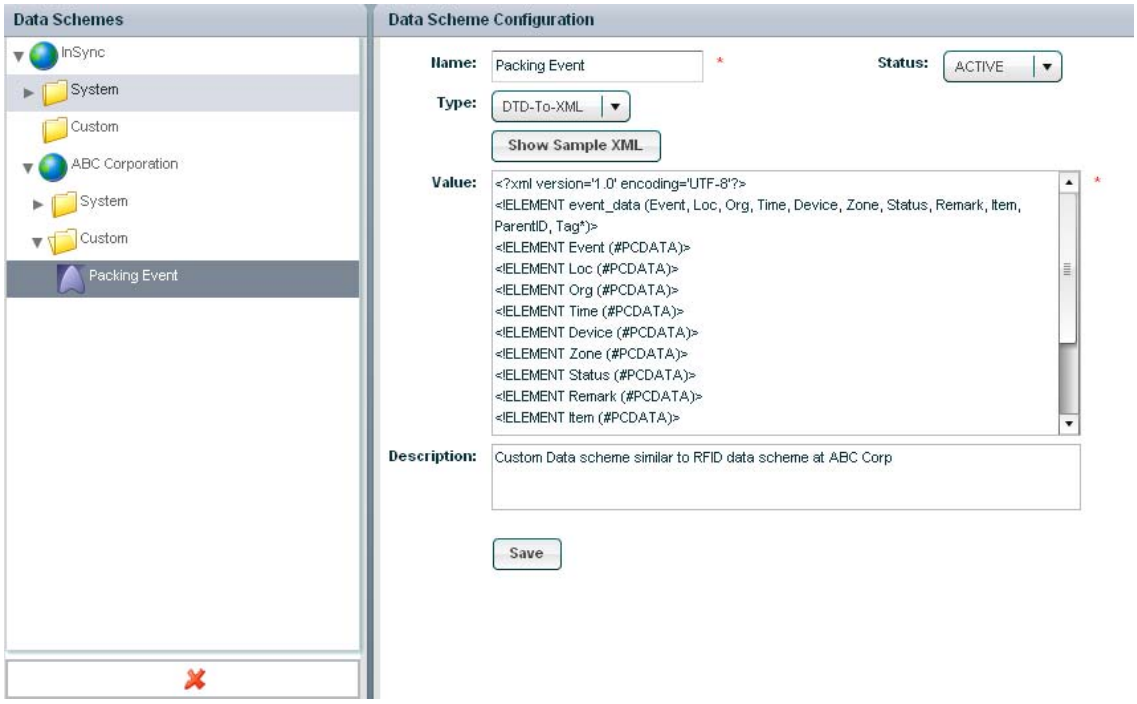
To create a Data Scheme

- Select the Organization
- Select System or Custom based on the access.
- Click



The attributes of the Data Scheme are

Attribute	Description
Name	Unique name within the Enterprise organization
Status	Active / InActive
Type	Type of the data scheme. Currently IApp supports DTD-XML conversion
Value	Is the DTD to be used for XML creation and validation
Description	Short description



Click ***Save*** to save the Data scheme. The Data scheme gets added to the left side tree.

4.15.1 Export / Import

Data Schemes provides Export/Import of data at the Home Enterprise level where all the data schemes gets exported / imported and at a particular Enterprise where only data schemes of that Enterprise are exported / imported.

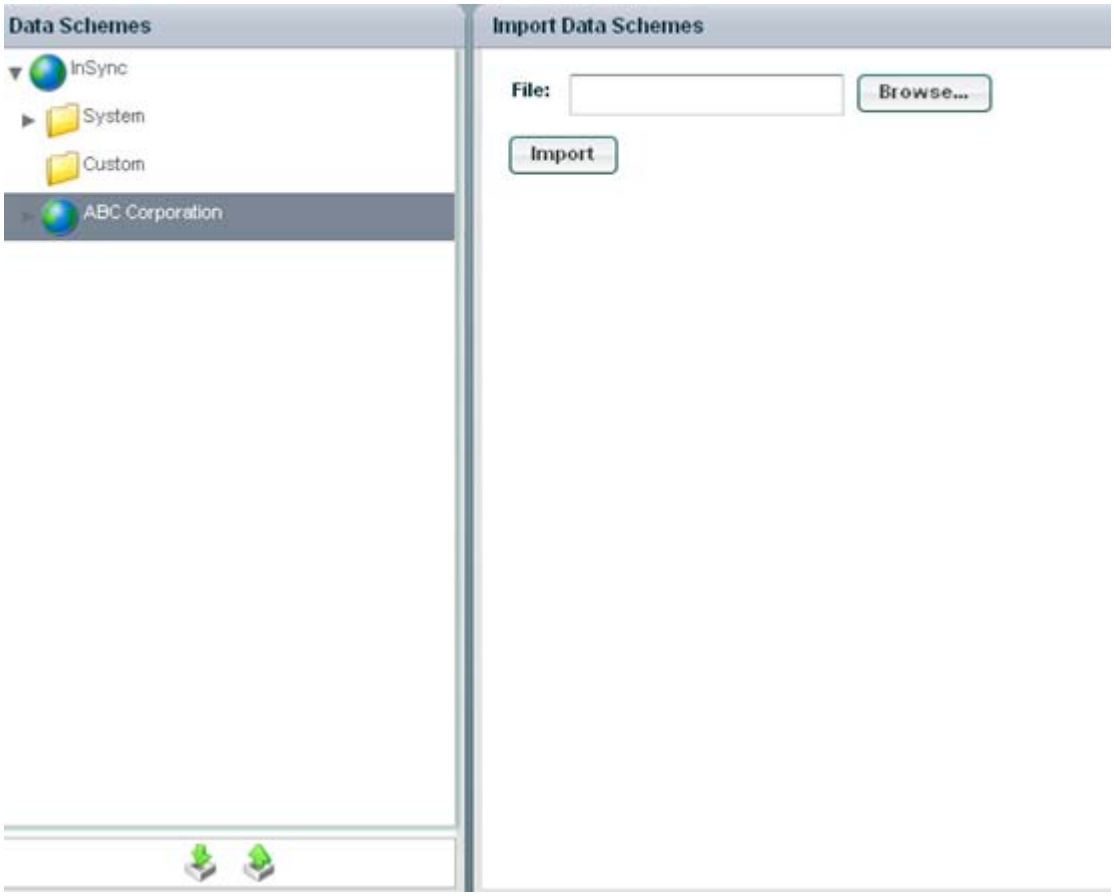
Export

- Select the Enterprise organization of interest and select to export the data schemes
- Select the location to save the file. The file dataschemes.xml format gets saved in the selected location.

Note: When export happens at the Home Enterprise it exports all the Enterprise information existing in the system.

Import

- Select the enterprise organization and select to import the file.



- Browse to the dataschemes.xml and click Import.
- Dataschemes gets imported into the system.

Pre-Requisites for Data Schemes Import

It is assumed that the following information already exists in IApp before the data schemes are imported.

- Enterprise Organization.
- Catalogs

If the above information does not exist then it is required that the above information is setup before the data schemes import is executed.

4.16 Formatters

Formatters are templates that are defined to format the events to be delivered to the subscribers and will contain element variables which are sourced from the Event Schema. It is responsible for converting the Event, Event Properties and Data into the final notification data format.

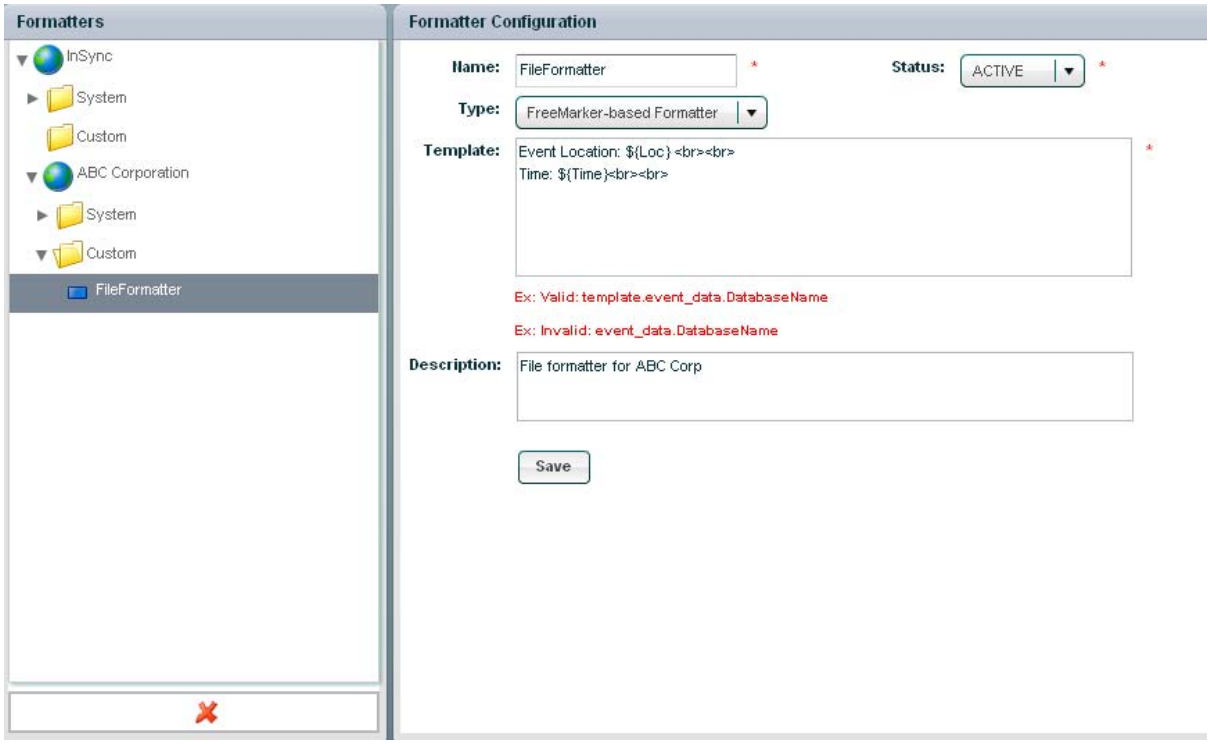
Formatters in IApp are listed by Enterprise – System / Custom.

- The System Formatters are provided by default and are available across all Enterprises.
- Only users with Administrative rights at the Home organization will be able to add or modify the System Formatters
- Users belonging to an Enterprise organization will be able to add new Formatters and modify the existing formatters in that Enterprise only.
- Administration users at Home Organization will be able to add custom Formatters in other Enterprise organizations.

4.16.1 Create Formatter

To create a Formatter

- Select the Organization
- Select System or Custom based on the access.
- Click



The attributes of the Formatter are

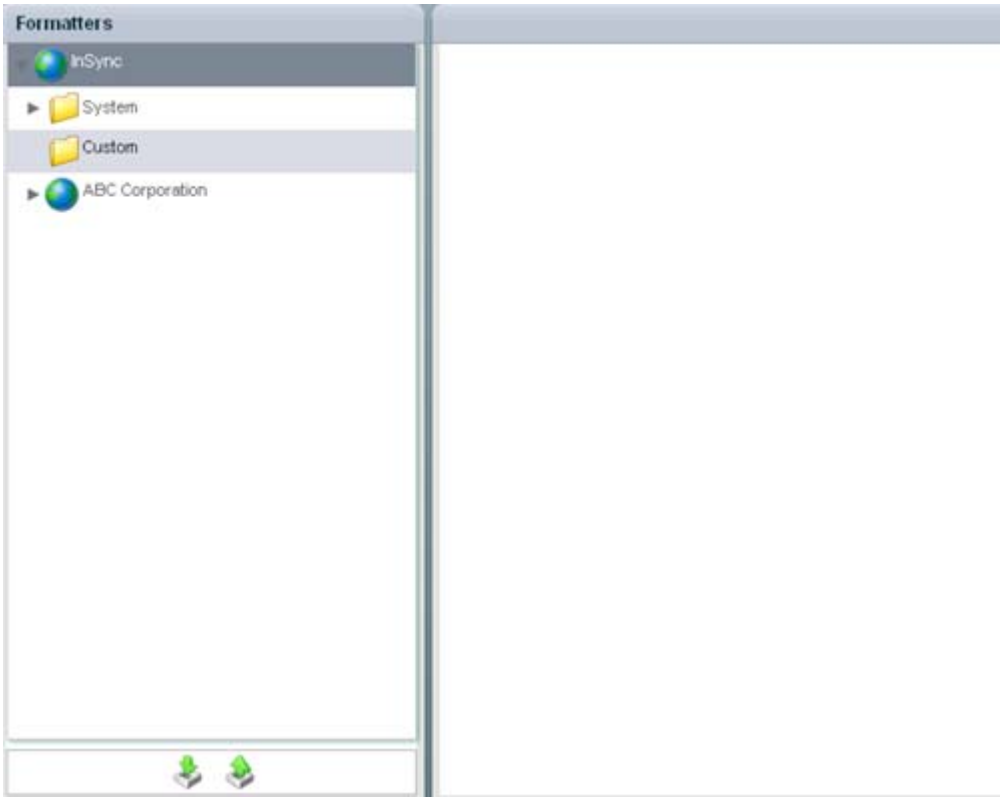
Attribute	Description
Name	Unique name within the Enterprise organization
Status	Active / InActive
Type	Type of Formatter. Freemarker-based Formatter / XSLT- based formatter. Freemarker – is a textual representation with variables which will be replaced during run time XSLT based – is an XSLT transformation based on the XSL stylesheet.
Template	is the format for the formatter. Based on the Type the template will have to be appropriately added. The template will be of free flowing text if it is of Freemarker based type The template will be of XSL stylesheet if is of XSLT based type
Description	Short description

Click ***Save*** to save the Formatter. The Formatter gets added to the left side tree panel.

Note: The variables used in the formatter {Loc} and {Time} are inputs to the formatter from the event. At the time of the event getting formatted these variables are replaced by the data from the event. For this conversion to be successful the variable names should match to the element names existing in the event.

4.16.2 Export / Import

Formatters provide Export/Import of data at the Home Enterprise level where all the data gets exported / imported and at a particular Enterprise where only formatters of that Enterprise are exported / imported.



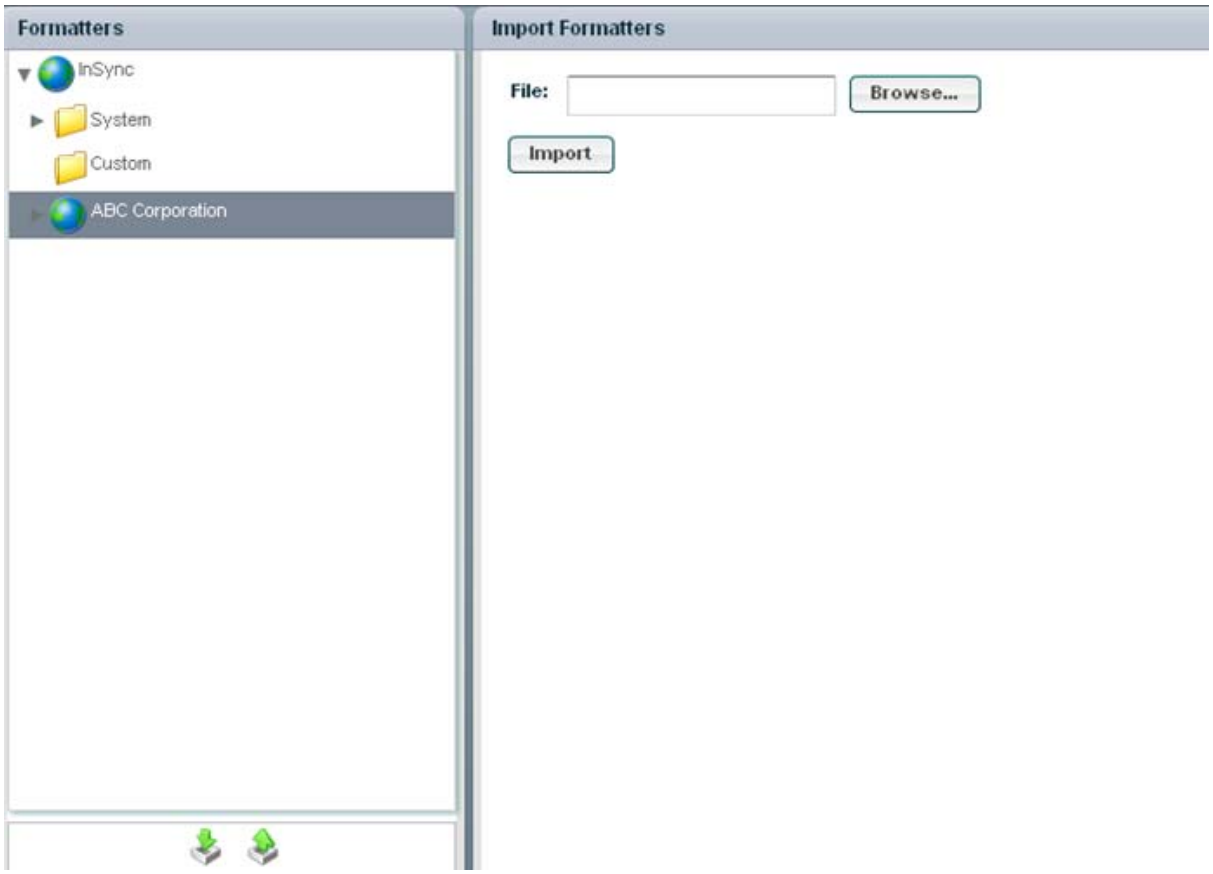
Export

- Select the Enterprise organization of interest and select to export the formatters
- Select the location to save the file. The file formatters.xml format gets saved in the selected location.

Note: When export happens at the Home Enterprise it exports all the Enterprise information existing in the system.

Import

- Select the enterprise organization and select to import the file.



- Browse to the formatters.xml and click Import.
- Formatters gets imported into the system.

Pre-Requisites for Formatters Import

It is assumed that the following information already exists in IApp before the formatters are imported.

- Enterprise Organization.
- Catalogs

If the above information does not exist then it is required that the above information is configured before the formatters are imported.

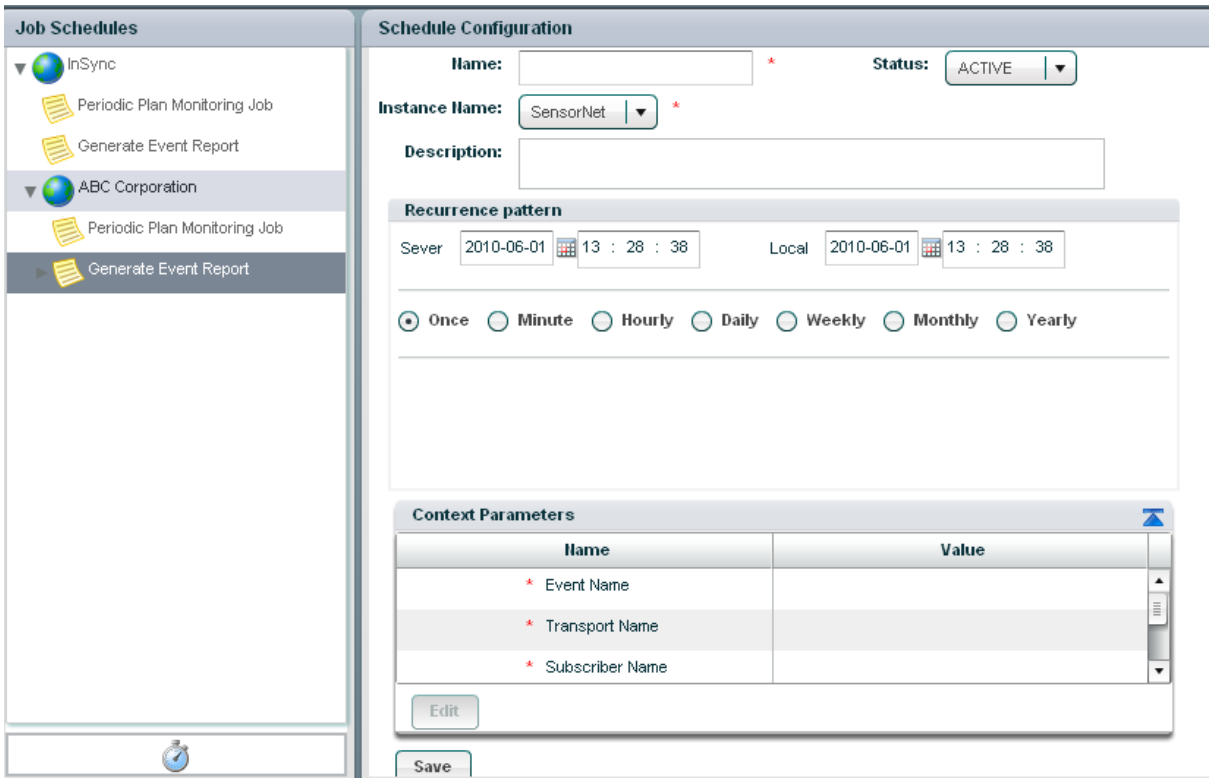
4.17 Schedule Jobs

Jobs are logical units with a specific implementation created for a purpose. The Jobs in IApp are created from the Software section and are scheduled here. Scheduling the job involves setting up the scheduler to run at the defined intervals.

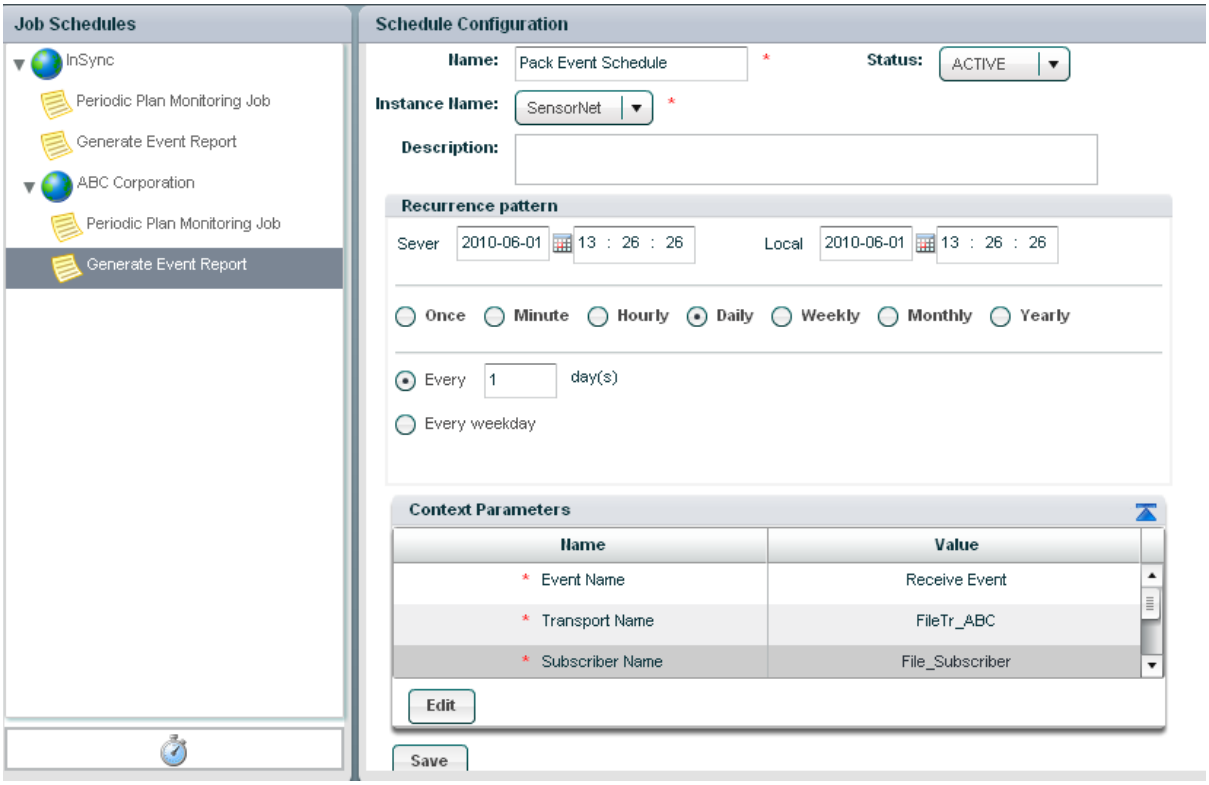
By default IApp provides the Generate Event Report and Periodic Plan Monitoring Job which can be scheduled for a particular Event of interest. The Jobs that are visible are created by Enterprise Organization from the Software menu item in the first level menu.

4.17.1 Create Schedule

Select the Job and Click on to create the Scheduler for the job.



Attribute	Description
Name	Scheduler name
Status	Active / InActive
Description	Short description
Recurrence Pattern	Date, time and frequency when the job has to run
Context Parameters	These are parameters which the job understands and is built into the job implementation. The values provided here are passed onto the Job Implementation.



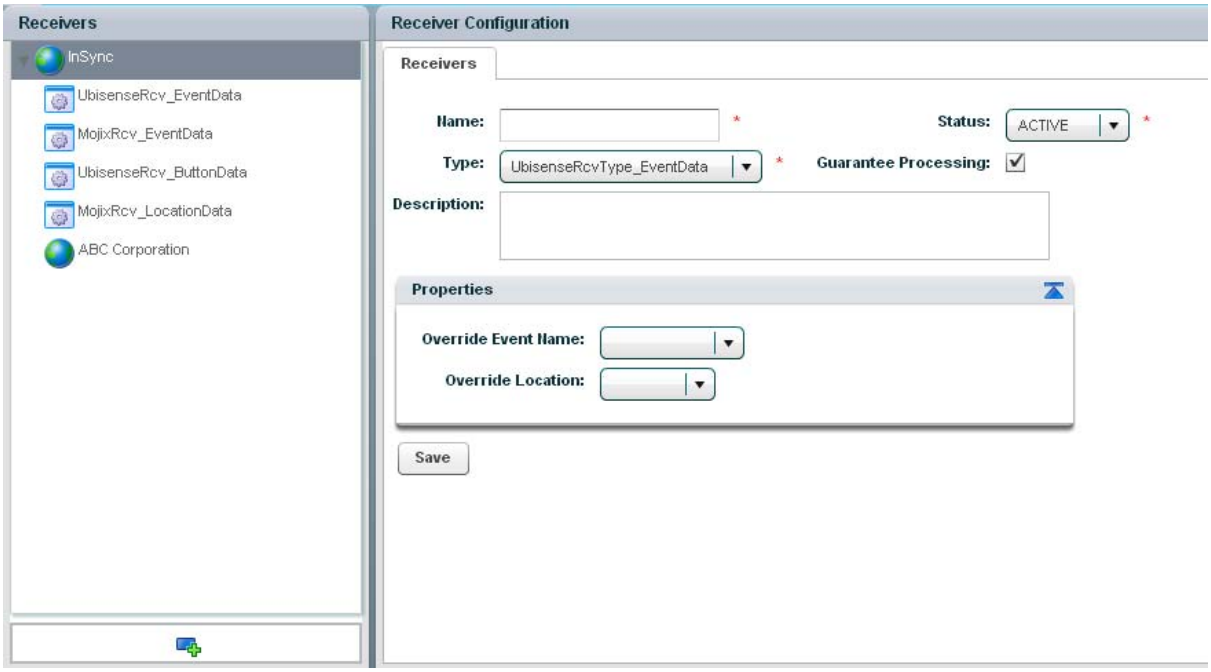
The Generate Event Report for which the job has been scheduled understands the following context parameters.

Context Parameter	Description
Event Name	Event name already defined in IApp.
Lot Size	No of events to be sent together at a time.
Subscriber Name	subscriber already defined in IApp to be used
Transport Name	Transport created for the Subscriber which will be used.

4.17.2 Receivers

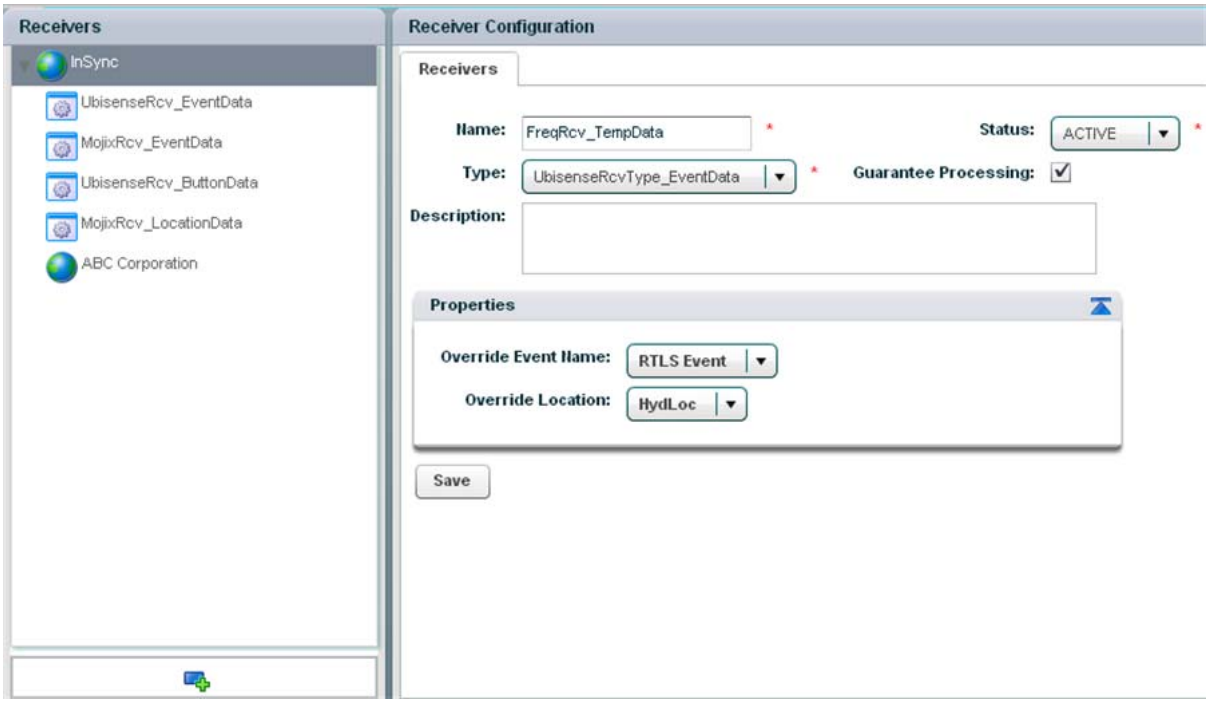
Receivers are a means of processing application data directly in sensornet wherein the need for a process zone and edge processes are not felt. This is mostly applicable to RTLS based systems and the like where data directly is sent to sensornet.

Each receiver takes the receiver type and the



Create Receiver

Click on to create a receiver



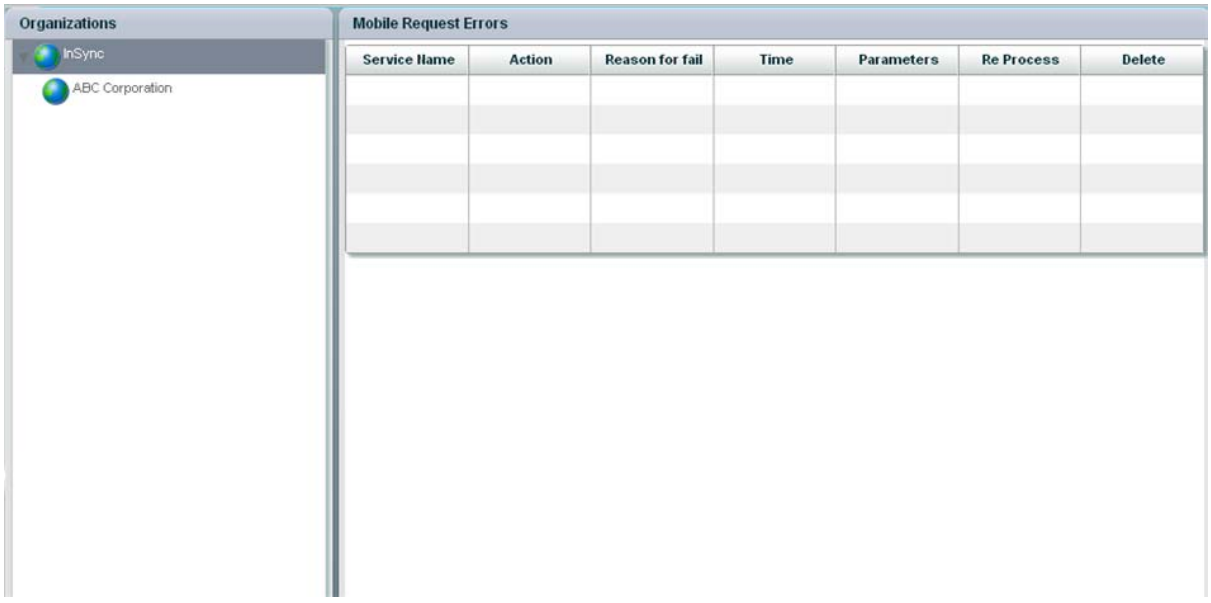
Attribute	Description
Name	Unique receiver name within the Enterprise organization. The external requests should have the receiver name in the request so that iApp can process the request.
Status	Active / InActive
Type	Type of the receiver. **Note:** Receiver types are created from the software menu->Receiver Types.
Guarantee Processing	Select the checkbox if guaranteed processing of the event is required.
Description	Short description
Properties	
Override Event Name	Event to be raised by the receiver. The listing is based on the receiver type configured on the receiver.
Override Location	Overrides the incoming location with the location selected in the final event.

- Click ‘Save’ to save the receiver.
- To delete a receiver select the receiver from the tree and click on .

Note: If the Type of the receiver selected contains the subscription class then the subscription tab becomes enabled after the receiver is created. This subscription is more to register the external system from where the requests will come. This feature is still not supported fully in this version.

4.17.3 Mobile Request Errors

Mobile requests to sensornet which are not processed successfully due to infrastructure related issues can be reprocessed in sensornet using the Mobile request errors page.

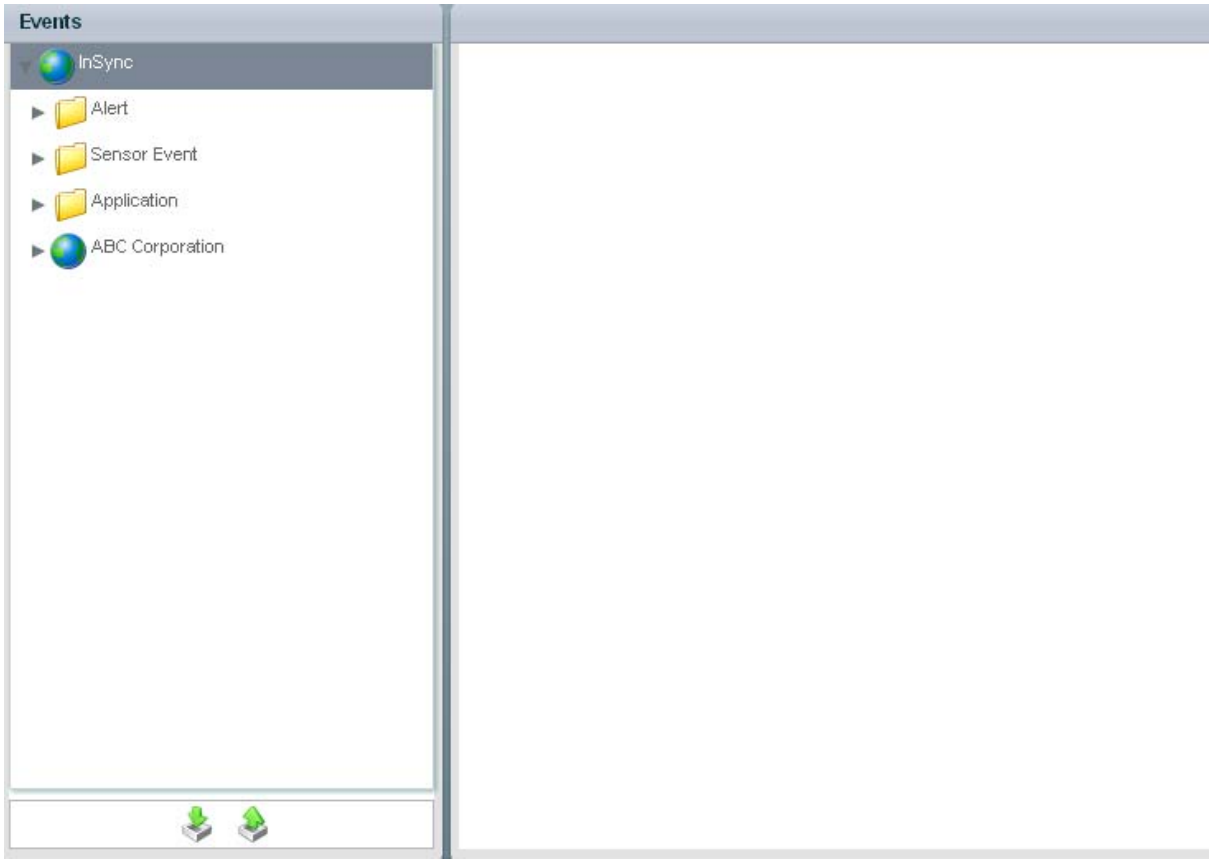


- The datagrid displays all failed requests in the order of failure.
- The Reprocess can be executed on the requests in the order they are listed. IOut of sequence reprocessing is not possible and cannot be executed.
- The failed requests can be deleted using the button in the delete column.

4.17.4 Export / Import

EMS provides Export/Import of data at the Home Enterprise level where the complete EMS gets exported / imported and at a particular Enterprise where only EMS of that Enterprise is exported / imported.

The feature is available from the EMS->Events tab.



Export

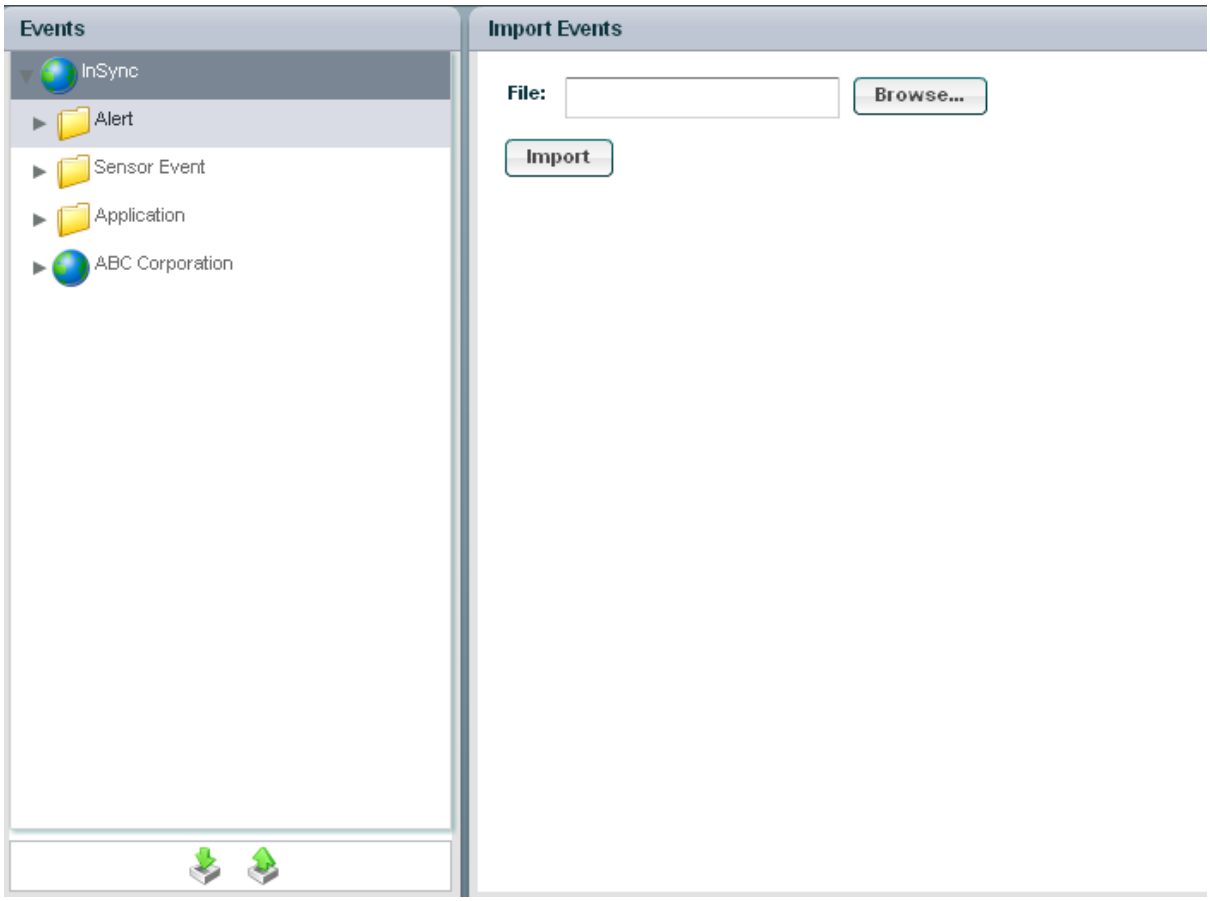
- Select the Enterprise organization of interest and select to export EMS.
- Select the location to save the file. The file ems.xml format gets saved in the selected location.
- Export saves the following entity information to the file
    - Events
    - Transports with the types
    - Data Schemes
    - Formatters with the types
    - Jobs

- Subscriptions
- Subscribers

Note: When export happens at the Home Enterprise it exports all the Enterprise information existing in the system.

Import

➢ Select the enterprise organization and select to import the file.



➢ Browse to the ems.xml and click Import.
➢ EMS data gets imported into the system.

Pre-Requisites for EMS Import

It is assumed that the following information already exists in IApp before EMS is imported.

- Enterprise Organization.
- Catalogs
- Location Configuration
- Event Handlers

If the above information does not exist then it is requires that the above information is configured before the EMS import is executed.

4.18 Monitoring

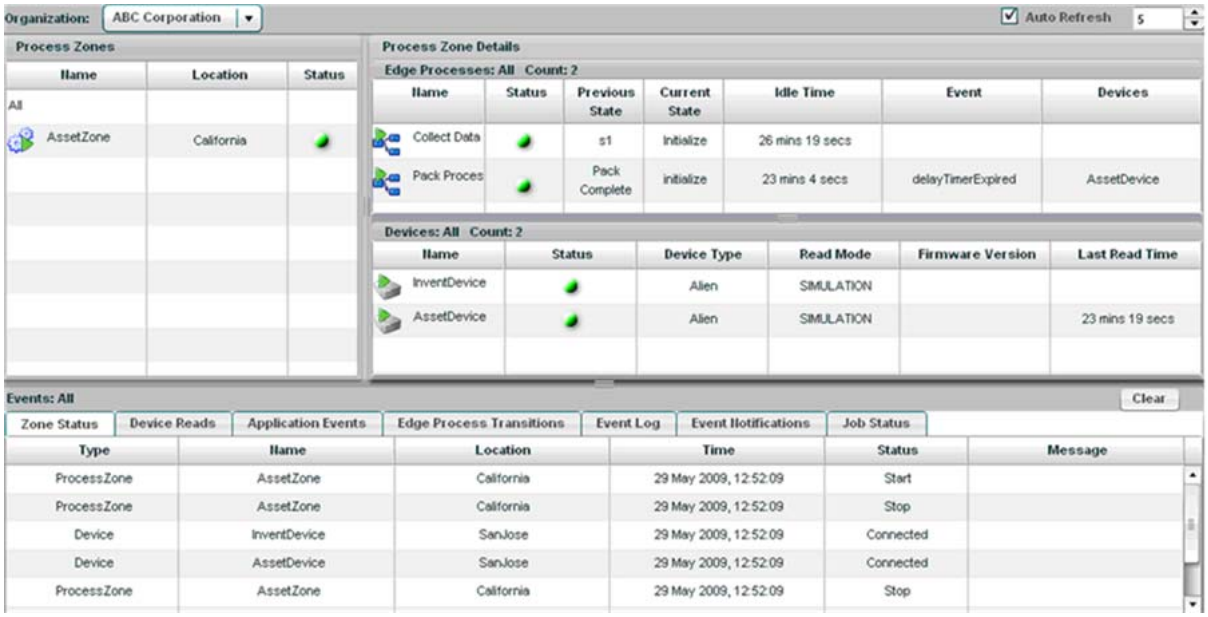
Monitoring in IApp provides a dashboard to monitor events, manage Process Zones, Devices and Edge processes by Enterprise organization. Monitoring is divided into 3 sections.

- Overview
  - Listing of the process zones, devices and edge processes within the zones with the ability to start/stop/reload them at any point of time.
  - Monitor Event Information, Edge Process Transitions and the status of the scheduled jobs.
- Realtime
  - Monitor tag reads realtime as and when they are happening in IApp.
- History
  - Provides the history of all device reads (raw reads)

4.18.1 Overview

The monitoring overview is displayed by Enterprise organization. Select the organization of interest to view the dashboard for that enterprise.

By default all the zone information and its corresponding Edge processes and devices are displayed.



- Select the Zone to view the data for the zone. This displays all the Edge processes and Devices configured for the zone
- Select the Edge Process OR the device to view the Event details of the selected element.

The status column of Zone / Edge Process / Device is represented by the following.

Status	Icon
Running (Green Icon)	

Stopped (Red Icon)	
Lately no response (Yellow Icon) For a Zone if any one of many devices configured is not running then the zone will be in this status.	

The operations possible are

Element	Operations	Description
Process Zone	Reload	Reloads all the configuration data. Zone should be reloaded every time the configuration changes.
Edge Processes	Reload	Reloads the edge process transitions and properties. The edge process should be reloaded every time the edge process definitions changes.
	Pause	Stops a running edge process to be able to start at a later time.
	Resume	Resumes the paused edge process
	Abort	Completely stops the edge process. This is required when the edge process is stuck at a state and unable to proceed to the next state. Abort will stop the cycle the edge process in to be able to start a new cycle.
Devices	Start Stop Reload	Starts the device selected Stops the selected device Reloads the devices with the configuration changes if any.

Note: It is recommended to switch off the auto refresh on the panel before proceeding with any of the above operations.

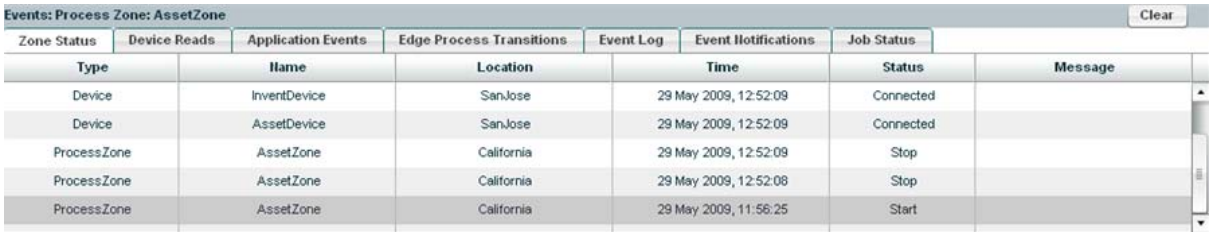
14.1.1 Events Panel:

The Events panel displays information of the events raised in the application along with the status of the zones, notifications, jobs and the edge process transitions.

14.1.2 Zone Status

The Zone Status tab displays the status (start / stop) of the zone and the devices within the zone.. This panel gives a view of the zone start and stop time, the device connection time along with the location.

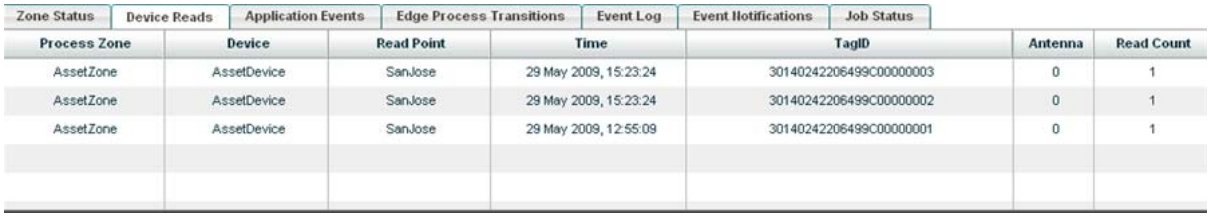
Selecting the device will display only the device details.



Type	Name	Location	Time	Status	Message
Device	InventDevice	SanJose	29 May 2009, 12:52:09	Connected	
Device	AssetDevice	SanJose	29 May 2009, 12:52:09	Connected	
ProcessZone	AssetZone	California	29 May 2009, 12:52:09	Stop	
ProcessZone	AssetZone	California	29 May 2009, 12:52:08	Stop	
ProcessZone	AssetZone	California	29 May 2009, 11:56:25	Start	

14.1.3 Device Reads

The Device reads displays the device reads as and when they are being read by the devices. The location, read point, read time and the tag id are displayed in the panel.



Process Zone	Device	Read Point	Time	TagID	Antenna	Read Count
AssetZone	AssetDevice	SanJose	29 May 2009, 15:23:24	30140242206499C00000003	0	1
AssetZone	AssetDevice	SanJose	29 May 2009, 15:23:24	30140242206499C00000002	0	1
AssetZone	AssetDevice	SanJose	29 May 2009, 12:55:09	30140242206499C00000001	0	1

To view the actual read double click on the row in the panel.

14.1.4 Application Events

Application Events are events raised by the Edge Process or by the handler configured. Click on ALL from the Process Zone panel to view the Application Events.

Location	Time	Event Name	Event Category
California	29 May 2009, 15:23:34	Pack Complete	Read
SanJose	29 May 2009, 15:23:24	Pack Complete	Read
California	29 May 2009, 12:55:20	Pack Complete	Read
SanJose	29 May 2009, 12:55:11	Pack Complete	Read

14.1.5 Edge Process Transitions

This displays all the edge process transitions as and when they are happening. Looking at the states will give a clear picture of whether the edge process state transitions are happening properly or not.

Select the edge process from the edge process panel to view the details of that edge process. Selecting the zone displays all the edge process information.

Edge Process Name	Location	Time	Previous State	Current State	Trigger Event
Pack Process at AssetZone	California	29 May 2009, 15:23:39	Pack Complete	initialize	delayTimerExpired
Pack Process at AssetZone	California	29 May 2009, 15:23:34	Building Pallet	Pack Complete	processFail
Pack Process at AssetZone	California	29 May 2009, 15:23:34	Pack Start	Building Pallet	readTimerExpired
Pack Process at AssetZone	California	29 May 2009, 15:23:24	initialize	Pack Start	tagRead
Pack Process at AssetZone	California	29 May 2009, 12:55:25	Pack Complete	initialize	delayTimerExpired

14.1.6 Event Log

The event log shows the status log of the event as it gets processed in IApp. At each process step it displays the unique ID which can be used for debug purposes.

UID	Event Name	Status	Time
100000142	Pack Complete	Processed	29 May 2009, 15:23:34
100000142	Pack Complete	To Be Processed	29 May 2009, 15:23:34
100000140	Pack Complete	Processed	29 May 2009, 15:23:24
100000140	Pack Complete	To Be Processed	29 May 2009, 15:23:24
100000139	Tag Read	Processed	29 May 2009, 15:23:24

14.1.7 Event Notifications

The event notification displays the delivery status of all the active subscriptions that are created for the event.

ID	Event Name	Transport	Subscriber	Status	Retry Count	Time
17039	Pack Complete	Mail_ABC	File_Subscriber	FAILED	3	29 May 2009, 15:23:42
17036	Pack Complete	Mail_ABC	File_Subscriber	FAILED	3	29 May 2009, 15:23:33
17033	Tag Read	File_ABC	File_Subscriber	DELIVERED	0	29 May 2009, 15:23:24
17028	Pack Complete	Mail_ABC	File_Subscriber	FAILED	3	29 May 2009, 12:55:29
17025	Pack Complete	Mail_ABC	File_Subscriber	FAILED	3	29 May 2009, 12:55:21

14.1.8 Job Status

The panel displays the status of the jobs running in IApp. It shows when the job ran as well as the next due run.

Job Name	Schedule Name	Status	Last Run Time	Next Run Time	Note
Generate Event Report	ABC Job	Running		2009-05-29 10:00:00	

Clear

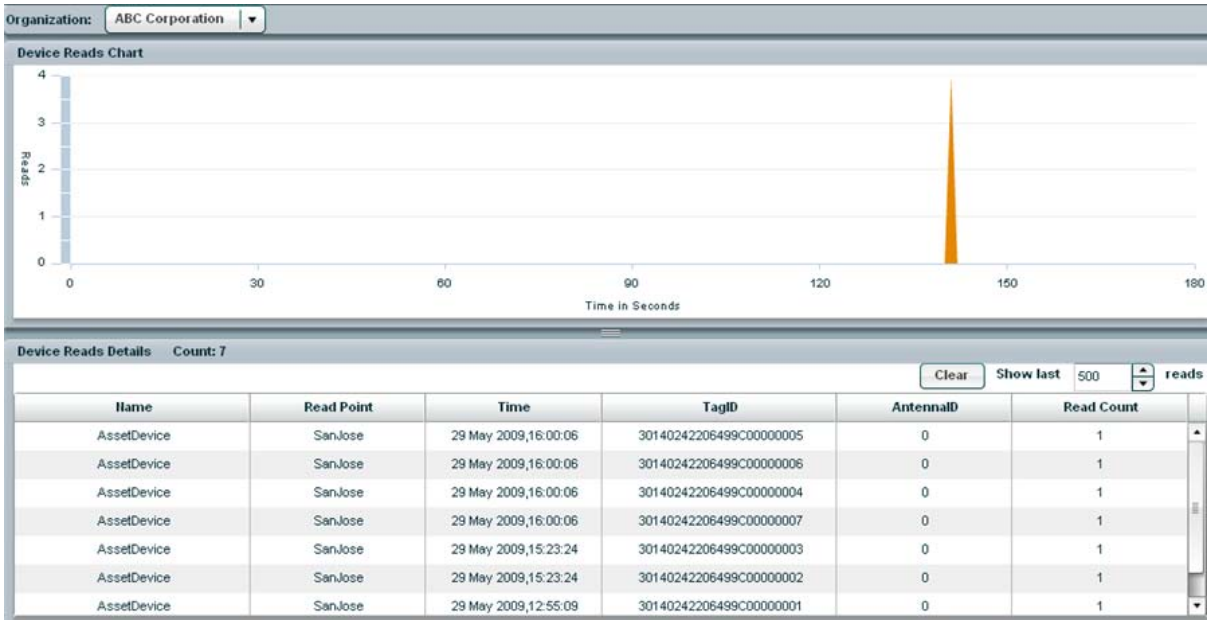
Click on clear to clear the contents of the active panel.

Note: 1. The data shown on each of the panel is the last 500 records for the respective information.
2. All Time columns shown in the panels are local browser times.

4.18.2 Realtime

The realtime tab is used to monitor the tag reads from all the devices within an Enterprise as and when they are happening.

Select the Enterprise organization from the drop down.


Organization: ABC Corporation
Device Reads Chart
Reads
Time in Seconds
Device Reads Details Count: 7
Clear
Show last 500 reads
Name | Read Point | Time | TagID | AntennaID | Read Count
AssetDevice | SanJose | 29 May 2009,16:00:06 | 30140242206499C00000005 | 0 | 1
AssetDevice | SanJose | 29 May 2009,16:00:06 | 30140242206499C00000006 | 0 | 1
AssetDevice | SanJose | 29 May 2009,16:00:06 | 30140242206499C00000004 | 0 | 1
AssetDevice | SanJose | 29 May 2009,16:00:06 | 30140242206499C00000007 | 0 | 1
AssetDevice | SanJose | 29 May 2009,15:23:24 | 30140242206499C00000003 | 0 | 1
AssetDevice | SanJose | 29 May 2009,15:23:24 | 30140242206499C00000002 | 0 | 1
AssetDevice | SanJose | 29 May 2009,12:55:09 | 30140242206499C00000001 | 0 | 1

- The number from the ‘Show last reads’ decides the number of reads that will get displayed in the grid.
- Clear button removes the data from the panel.

4.18.3 History

The history panel displays the device raw reads by time or by device.

- Select the Enterprise organization from the drop down
- Select the Zone and the devices.
- Click Refresh

4.18.4 By Time

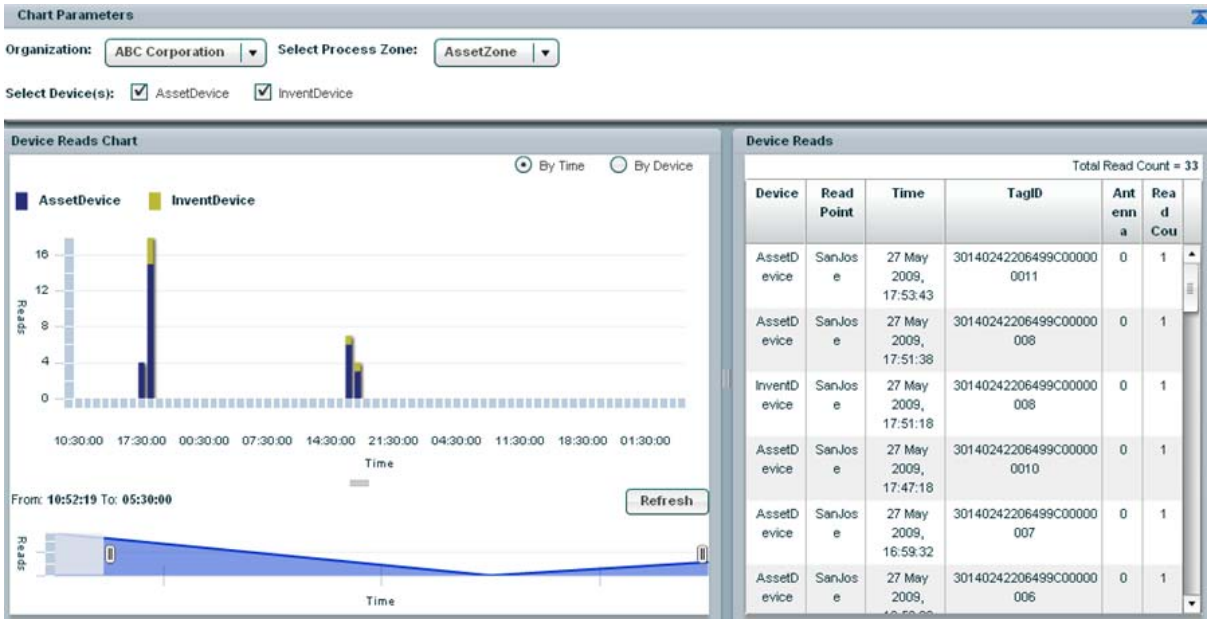
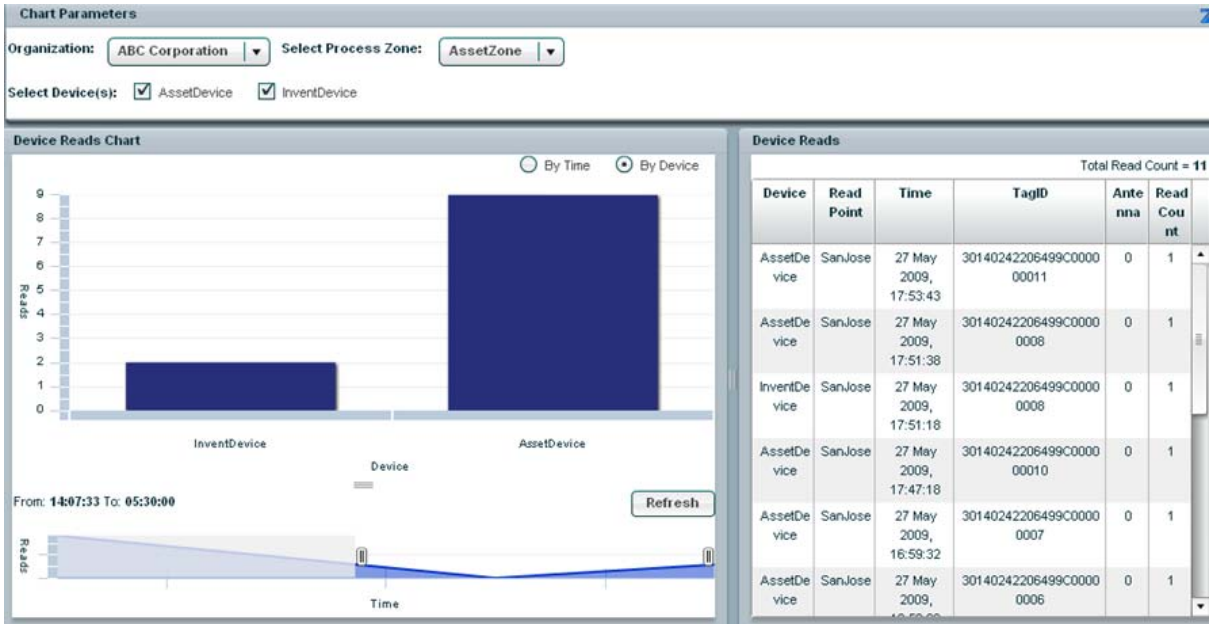


Chart Parameters
Organization: ABC Corporation
Select Process Zone: AssetZone
Select Device(s): AssetDevice InventDevice
Device Reads Chart
By Time By Device
AssetDevice InventDevice
Reads
Time
From: 10:52:19 To: 05:30:00
Refresh
Device Reads
Total Read Count = 33
Device | Read Point | Time | TagID | Antenna | Read Cou
AssetDevice | SanJose | 27 May 2009, 17:53:43 | 30140242206499C00000011 | 0 | 1
AssetDevice | SanJose | 27 May 2009, 17:51:38 | 30140242206499C00000008 | 0 | 1
InventDevice | SanJose | 27 May 2009, 17:51:18 | 30140242206499C00000008 | 0 | 1
AssetDevice | SanJose | 27 May 2009, 17:47:18 | 30140242206499C00000010 | 0 | 1
AssetDevice | SanJose | 27 May 2009, 16:59:32 | 30140242206499C00000007 | 0 | 1
AssetDevice | SanJose | 27 May 2009, | 30140242206499C00000006 | 0 | 1

4.18.5 By Device



The slider control can be adjusted to select the time interval between which the reads will be displayed on the Device reads chart and the device reads table.

For further filtering select the bar in the device reads chart to view reads for that particular device or time, based on the view.

4.19 Alerts

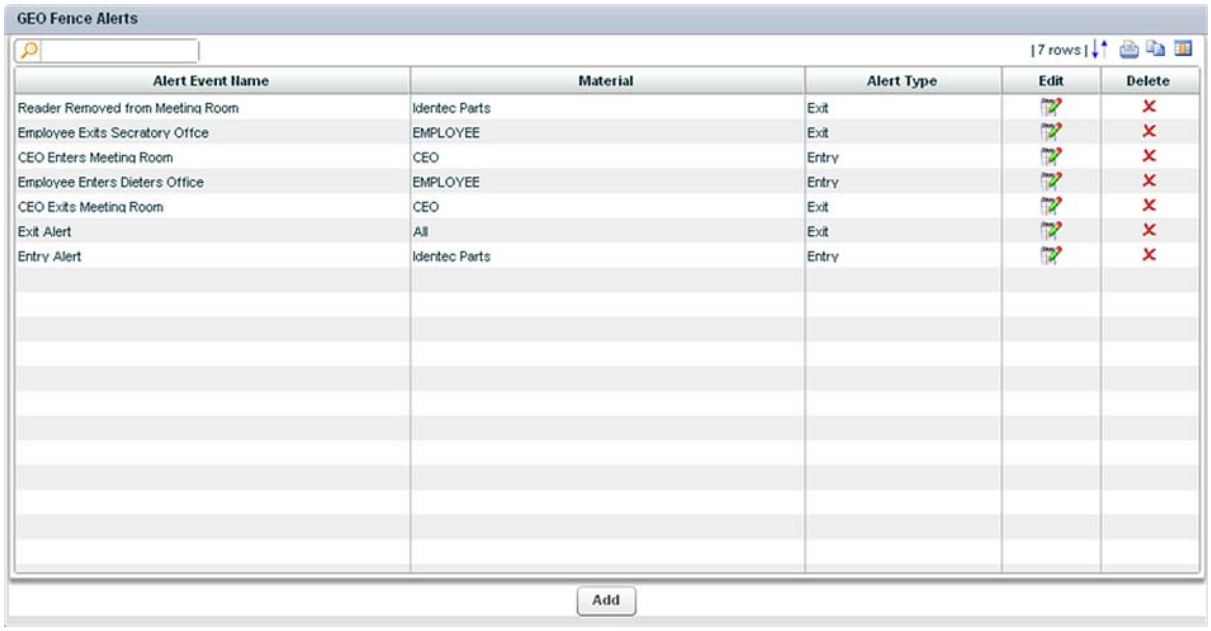
Click on to view the Alert menu.

Note:

1. Alerts created for alert configuration are created in the Enterprise organization of the logged in user.
2. All items, materials, locations and zones displayed in the panels are all of the enterprise of the logged in user.

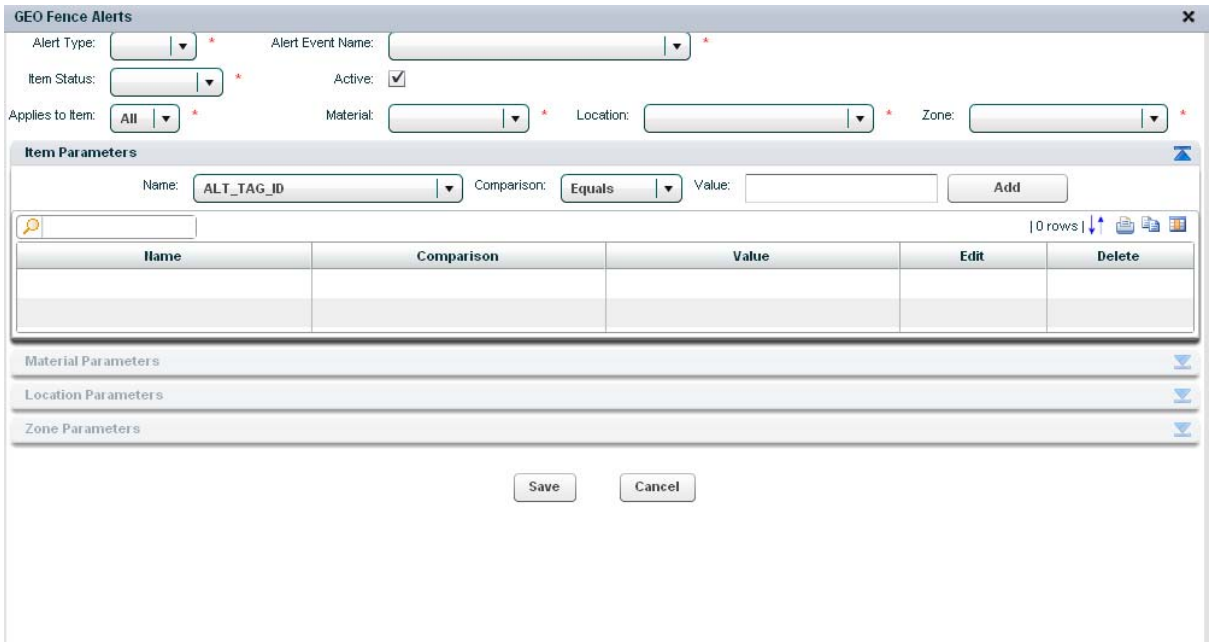
4.19.1 Geo Fence Alerts

Alerts configured around a geographical area are Geo Fence Alerts.



The grid displays all existing Geo Fence alerts configured.

- Click on ***Add*** button to create a new alert.

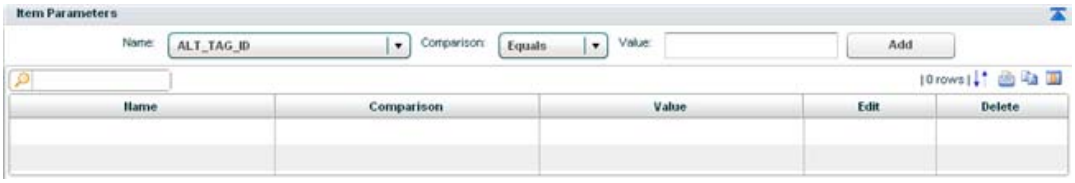


Attribute	Description
Alert Type	Defines the alert type – Entry or Exit
Alert Event Name	Events of type ‘Alert’ defined in the system
Item Status	Status of the item – Active/InActive
Active	Status of the geo fence alert created

Applies to	
Item	Item for which the alert is applicable. ‘All’ is applicable to all items moving in/out of the zones.
Material	Material for which the alert is applicable. ‘All’ is applicable to all material moving in/out of the zones.
Location	Location for which the alert is applicable. ‘All’ is applicable to all Locations moving in/out of the zones. The drop down lists all locations from the location hierarchy.
Zone	Zone for which the alert is applicable. ‘All’ is applicable to all zones. The drop down lists all location regions defined while calibrating the image.

Item Parameters

Panel is enabled only when ‘All’ is selected from the Item drop down. Use this for further filtering to raise the alert on the item by adding alternate attributes of the item to the configuration.



- This allows defining criteria for an attribute of the item other than the Item Id by selecting the attribute from the ***Name*** drop down.
- The comparison is the operator to compare the attribute to a static value. It supports Equals and Not Equals in this version.
- Click on ‘Add’ to add the item parameter

Material Parameters

Panel is enabled only when ‘All’ is selected from the Material drop down. Use this for further filtering to raise the alert on the Material by adding alternate attributes of the Material to the configuration.

- This allows defining criteria for an attribute of the Material other than the Material name by selecting the attribute from the ***Name*** drop down.
- The comparison is the operator to compare the attribute to a static value. It supports Equals and Not Equals in this version.
- Click on ‘Add’ to add the Material parameter

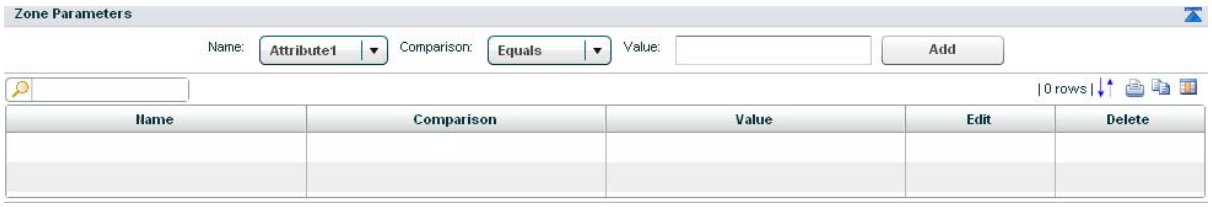
Location Parameters

Panel is enabled only when ‘All’ is selected from the Location drop down. Use this for further filtering to raise the alert on the Location by adding alternate attributes of the Location to the configuration.

- This allows defining criteria for an attribute of the Material other than the Material name by selecting the attribute from the ***Name*** drop down.
- The comparison is the operator to compare the attribute to a static value. It supports Equals and Not Equals in this version.
- Click on 'Add' to add the Material parameter.

Zone Parameters

Panel is enabled only when 'All' is selected from the Zone drop down. Use this for further filtering to raise the alert on the Zone by adding alternate attributes of the Zone to the configuration.



- This allows defining criteria for an attribute of the Zone other than the Zone name by selecting the attribute from the ***Name*** drop down.
- The comparison is the operator to compare the attribute to a static value. It supports Equals and Not Equals in this version.
- Click on 'Add' to add the Zone parameter.

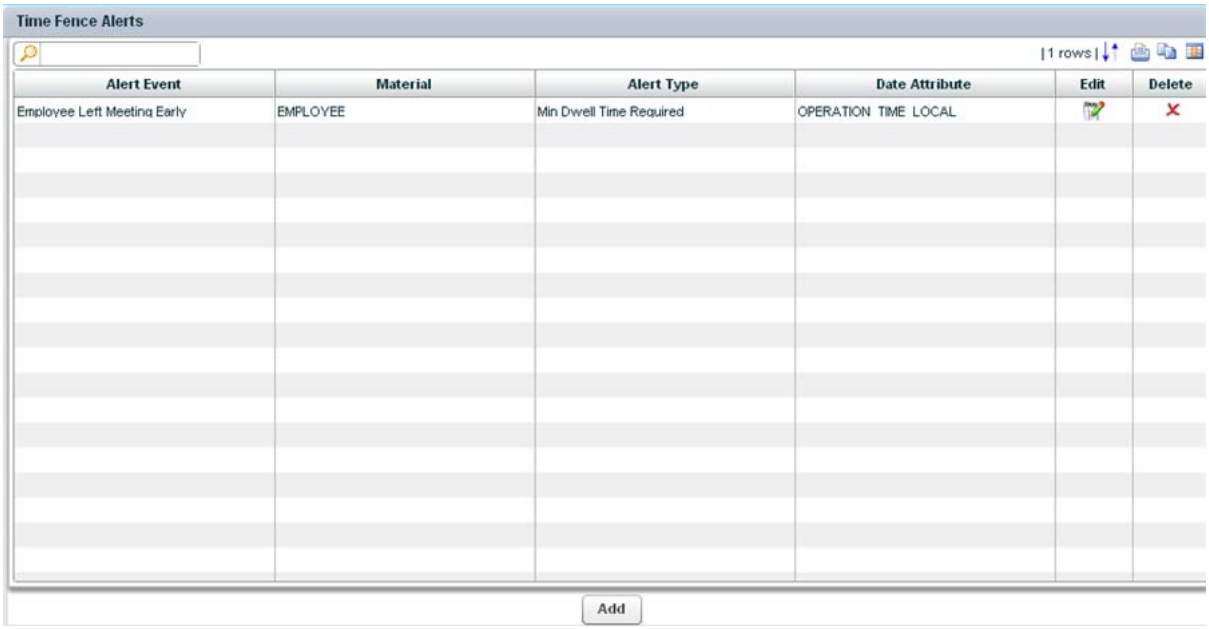
- Click to edit the alert.
- Click to delete the alert.

Note:

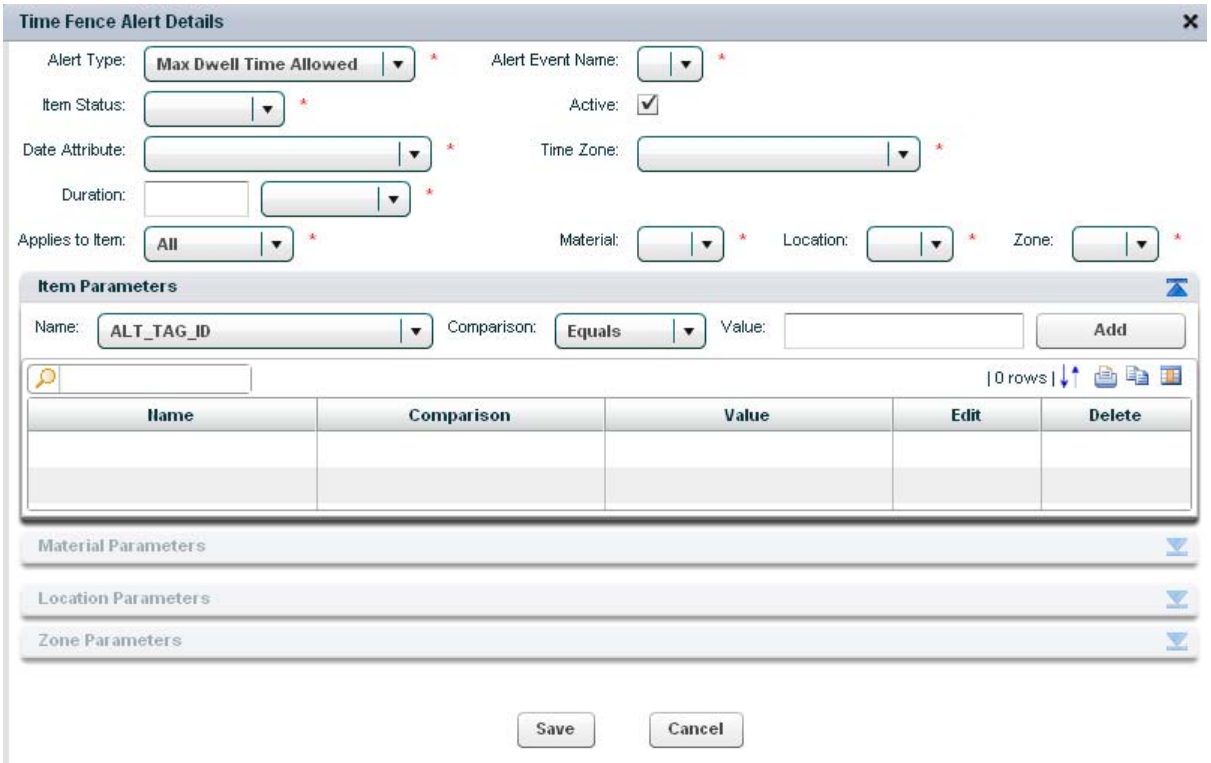
1. Items, Material, Locations and Zones have to be created for Geo alerts to be configured on them.
2. The Geo Fence Alerts (Entry/Exit) are raised based on the combination of the Item, Material, Location and Zone attributes set. It is an 'AND' and not an 'OR' operation.
3. RTLs event handler should be configured for the Entry/Exit alerts to be raised.

4.19.2 Time Fence Alerts

Alerts configured around boundaries between different time periods are Time fence alerts.



- Click on ***Add*** button to create a new alert.



Attribute	Description
Alert Type	Defines the alert type – Max Dwell time allowed / Min Dwell time required. Max dwell time is the maximum time that an asset can reside in a designated zone. The alert is raised if the asset stays beyond the

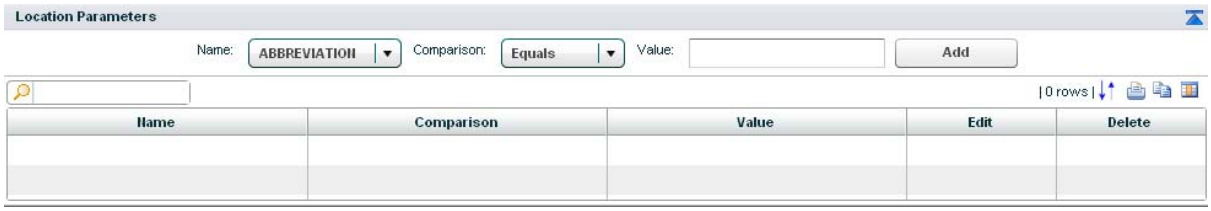
	defined time in the zone. Min dwell time is the minimum time that an asset should reside in a designated area. The alert is raised if the asset moves out of the zone within the defined time.
Alert Event Name	Events of type ‘Alert’ defined in the system
Item Status	Status of the item – Active/InActive
Active	Status of the time fence alert created
Date Attribute	Date attribute of the asset that has to be monitored for the alerts. When the value of the selected Date Attribute changes, application compares the difference between the value in the selected Date Attribute and current system time and the Duration mentioned in the Alert Configuration. If the difference is more than the Duration, ‘Max Dwell Time Allowed’ alert will be raised. If the difference is less than the Duration, ‘Min Dwell Time Required’ alert will be raised.
Time Zone	Time zone of the selected date attribiute
Duration	Time set for the alerts to be raised.
Applies to	
Item	Item for which the alert is applicable. ‘All’ is applicable to all items moving in/out of the zones.
Material	Material for which the alert is applicable. ‘All’ is applicable to all material moving in/out of the zones.
Location	Location for which the alert is applicable. ‘All’ is applicable to all Locations moving in/out of the zones. The drop down lists all locations from the location hierarchy.
Zone	Zone for which the alert is applicable. ‘All’ is applicable to all zones. The drop down lists all location regions defined while calibrating the image.
Item Parameters Panel is enabled only when ‘All’ is selected from the Item drop down. Use this for further filtering to raise the alert on the item by adding alternate attributes of the item to the configuration. • This allows defining criteria for an attribute of the item other than the Item Id by selecting the attribute from the ***Name*** drop down. • The comparison is the operator to compare the attribute to a static value. It supports Equals and Not Equals in this version. • Click on ‘Add’ to add the item parameter	
Material Parameters Panel is enabled only when ‘All’ is selected from the Material drop down. Use this for further filtering	

to raise the alert on the Material by adding alternate attributes of the Material to the configuration.

- This allows defining criteria for an attribute of the Material other than the Material name by selecting the attribute from the ***Name*** drop down.
- The comparison is the operator to compare the attribute to a static value. It supports Equals and Not Equals in this version.
- Click on 'Add' to add the Material parameter

Location Parameters

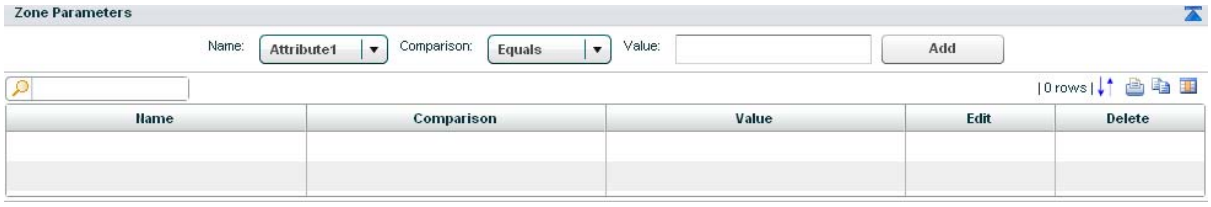
Panel is enabled only when 'All' is selected from the Location drop down. Use this for further filtering to raise the alert on the Location by adding alternate attributes of the Location to the configuration.



- This allows defining criteria for an attribute of the Material other than the Material name by selecting the attribute from the ***Name*** drop down.
- The comparison is the operator to compare the attribute to a static value. It supports Equals and Not Equals in this version.
- Click on 'Add' to add the Material parameter.

Zone Parameters

Panel is enabled only when 'All' is selected from the Zone drop down. Use this for further filtering to raise the alert on the Zone by adding alternate attributes of the Zone to the configuration.



- This allows defining criteria for an attribute of the Zone other than the Zone name by selecting the attribute from the ***Name*** drop down.
- The comparison is the operator to compare the attribute to a static value. It supports Equals and Not Equals in this version.
- Click on 'Add' to add the Zone parameter.

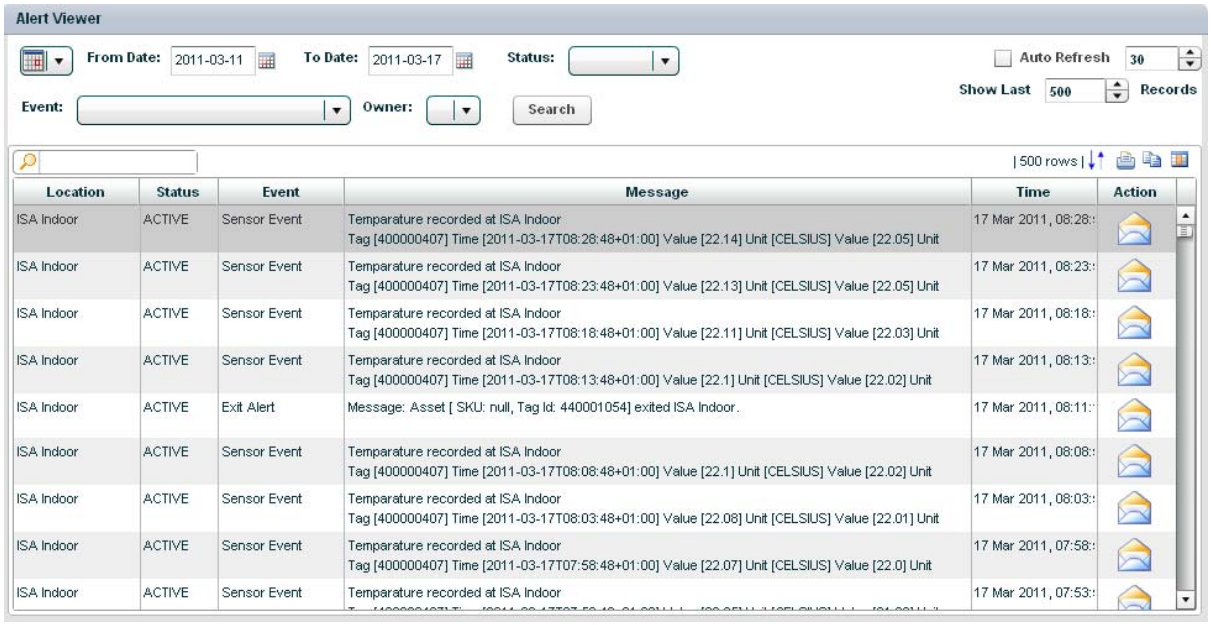
- Click to edit the alert.
- Click to delete the alert.

Note:

- Items, Material, Locations and Zones have to be created for Time fence alerts to be configured on them.
- The Time Fence Alerts are raised based on the combination of the Item, Material, Location and Zones attributes set. It is an ‘AND’ and not an ‘OR’ operation.
- RTLS Event handler should be configured in EMS for the Min Dwell Time alert to be raised.
- The Timed Alert monitor job should be scheduled for the Max Dwell time allowed alert to be raised.

4.19.3 Alert Viewer

Alert viewer displays all the alerts that are raised in the system for appropriate action. The search enables the filtering of the alerts based on the criteria provided.


Alert Viewer
From Date: 2011-03-11
To Date: 2011-03-17
Status:
Auto Refresh 30
Show Last 500 Records
Event:
Owner:
Search
500 rows
Location | Status | Event | Message | Time | Action
ISA Indoor | ACTIVE | Sensor Event | Temparature recorded at ISA Indoor Tag [400000407] Time [2011-03-17T08:28:48+01:00] Value [22.14] Unit [CELSIUS] Value [22.05] Unit | 17 Mar 2011, 08:28:
ISA Indoor | ACTIVE | Sensor Event | Temparature recorded at ISA Indoor Tag [400000407] Time [2011-03-17T08:23:48+01:00] Value [22.13] Unit [CELSIUS] Value [22.05] Unit | 17 Mar 2011, 08:23:
ISA Indoor | ACTIVE | Sensor Event | Temparature recorded at ISA Indoor Tag [400000407] Time [2011-03-17T08:18:48+01:00] Value [22.11] Unit [CELSIUS] Value [22.03] Unit | 17 Mar 2011, 08:18:
ISA Indoor | ACTIVE | Sensor Event | Temparature recorded at ISA Indoor Tag [400000407] Time [2011-03-17T08:13:48+01:00] Value [22.1] Unit [CELSIUS] Value [22.02] Unit | 17 Mar 2011, 08:13:
ISA Indoor | ACTIVE | Exit Alert | Message: Asset [SKU: null, Tag Id: 440001054] exited ISA Indoor. | 17 Mar 2011, 08:11:
ISA Indoor | ACTIVE | Sensor Event | Temparature recorded at ISA Indoor Tag [400000407] Time [2011-03-17T08:08:48+01:00] Value [22.1] Unit [CELSIUS] Value [22.02] Unit | 17 Mar 2011, 08:08:
ISA Indoor | ACTIVE | Sensor Event | Temparature recorded at ISA Indoor Tag [400000407] Time [2011-03-17T08:03:48+01:00] Value [22.08] Unit [CELSIUS] Value [22.01] Unit | 17 Mar 2011, 08:03:
ISA Indoor | ACTIVE | Sensor Event | Temparature recorded at ISA Indoor Tag [400000407] Time [2011-03-17T07:58:48+01:00] Value [22.07] Unit [CELSIUS] Value [22.0] Unit | 17 Mar 2011, 07:58:
ISA Indoor | ACTIVE | Sensor Event | Temparature recorded at ISA Indoor | 17 Mar 2011, 07:53:

User can search alerts on the following criteria:

- From Date and To Date: This is a mandatory search criterion. All alerts fired within these dates and on these dates will be displayed in the viewer.
- Status: Active / InActive. User can search for only Active alerts or only Inactive alerts also.
- Event: The dropdown displays all the alerts that have been fired based on the alert configuration.
- Owner: Alert owners are owners of the asset. When a new asset is created, an owner name can be specified. The owner name should be a user of the application.
- The Show last record is the no of alerts to be displayed in the viewer.
- Auto refresh can be turned on or off by checking or unchecking the Auto refresh checkbox.
- Click on the in the Action column for the alert. It will display a popup for the user to enter a reason.
- The alert cannot be closed without a reason mentioned.

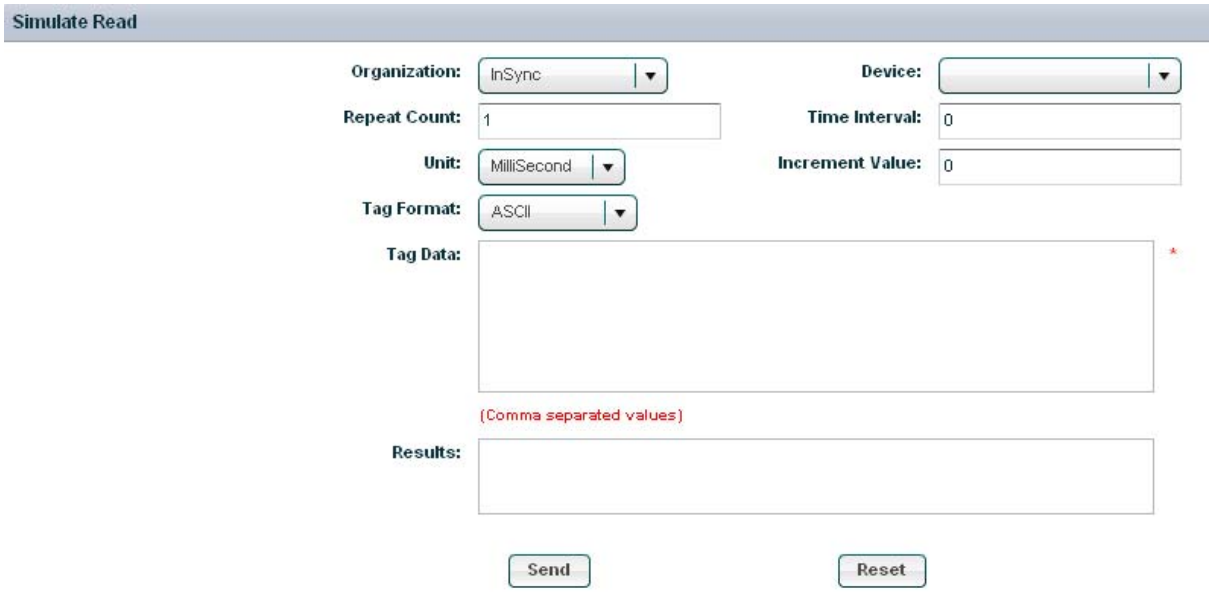
Note: The Alert event handler should be configured for all alerts to be displayed on the Alert viewer.

4.20 Activities

Activities are a set of utility programs for simulation purposes. It also provides GEN2 capabilities in working with RFID tags.

4.20.1 Simulate Read

To test the business logic/implementation before going for the entire setup with the hardware, use the Simulate Read function to replicate a scenario of reading a tag using a device.



The attributes are

Attributes	Description
Organization	The enterprise organization within which the devices are configured.
Devices	Devices configured for the enterprise.
Repeat Count	Number of times to repeat the simulation.
Time Interval	Duration between two simulations.
Unit	Unit of measurement of the time between two simulations. It may be Milliseconds or Seconds.
Increment Value	Value by which to increment the tag data after each simulation.
Tag Format	Format in which tag is being sent. It may be any one of the following: Hexadecimal, ASCII, Binary or Digit.
Tag Data	Enter tag id separated by comma to simulate multiple tag ids
Result	Displays simulation output. Success/Failed to simulate read.

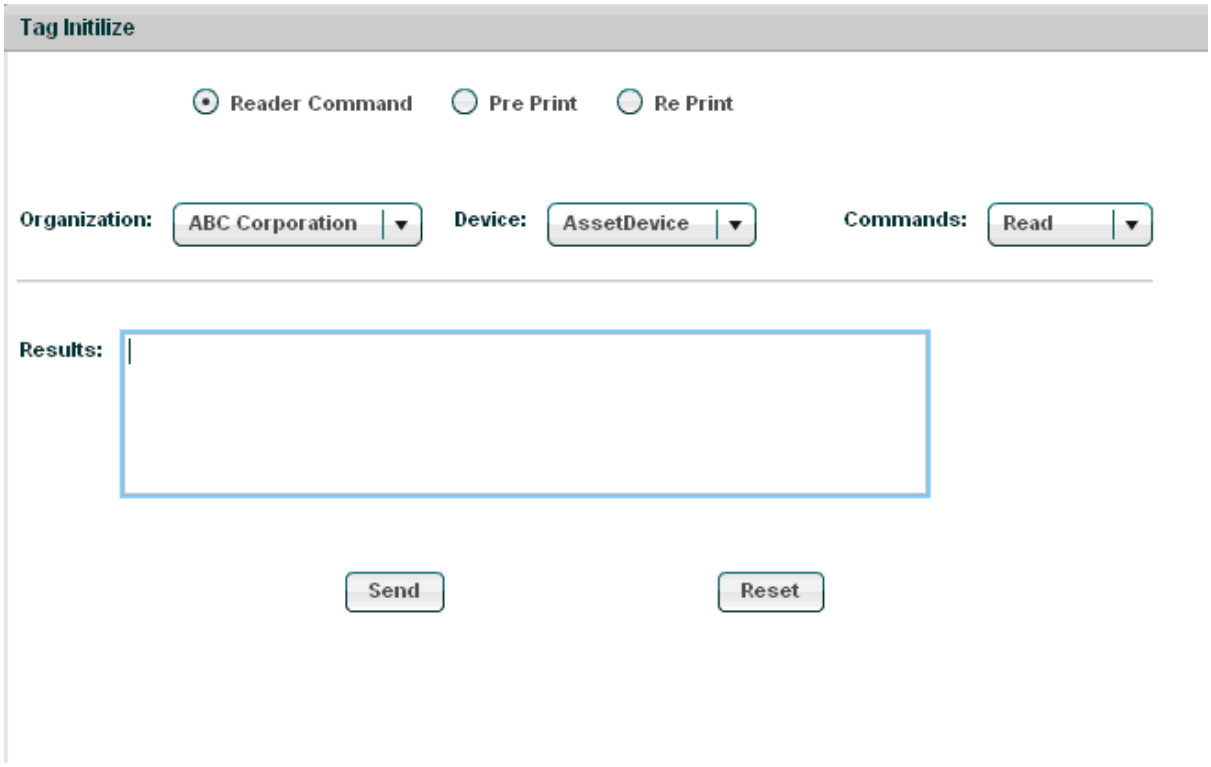
- The Result of the simulation is shown in the Result Field.
- If the device successfully reads the tag data, it displays "**Success**" in the result field, else it displays failure message along with the reason for failure.
- To simulate a set of tags provide the starting tag ID, the increment value and set the number of times to repeat the simulation and click Send.

4.20.2 Device Reads

Device Reads or Tag Initialize is to write tag ID to the current tag exposed to the device. It is also called tag commissioning. The tags can be read using the "Reader Command" or write Tag ID on the tag using the "Pre-Print" option or re-write Tag IDs on tag which could not be done properly on first attempt using the "Re Print" option.

4.20.3 Reader Command

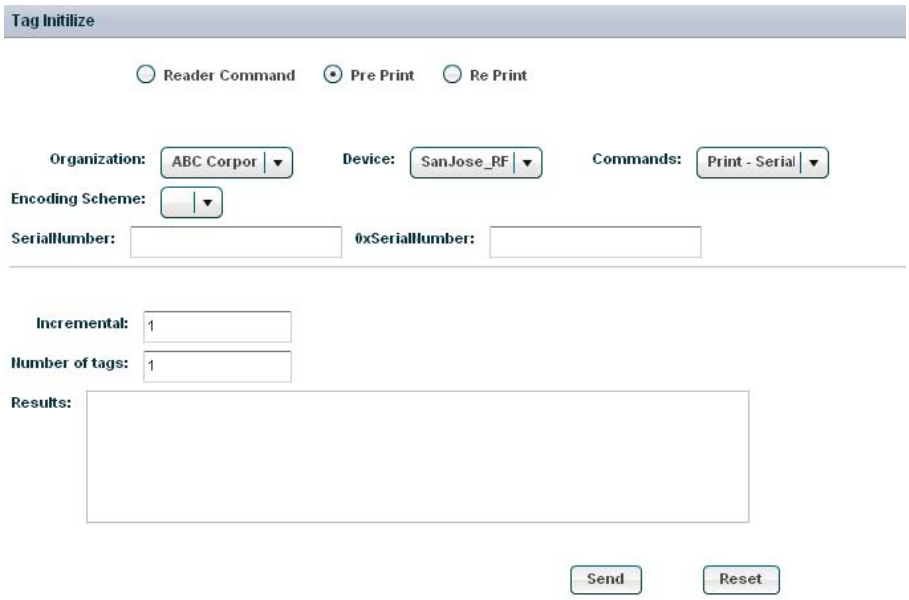
The commands associated with the reader can be executed using the reader command.



- Select the Reader Command radio button
- Select the Enterprise Organization
- Select the device and the command to be sent to the device
- Click 'Send'

4.20.4 Pre Print

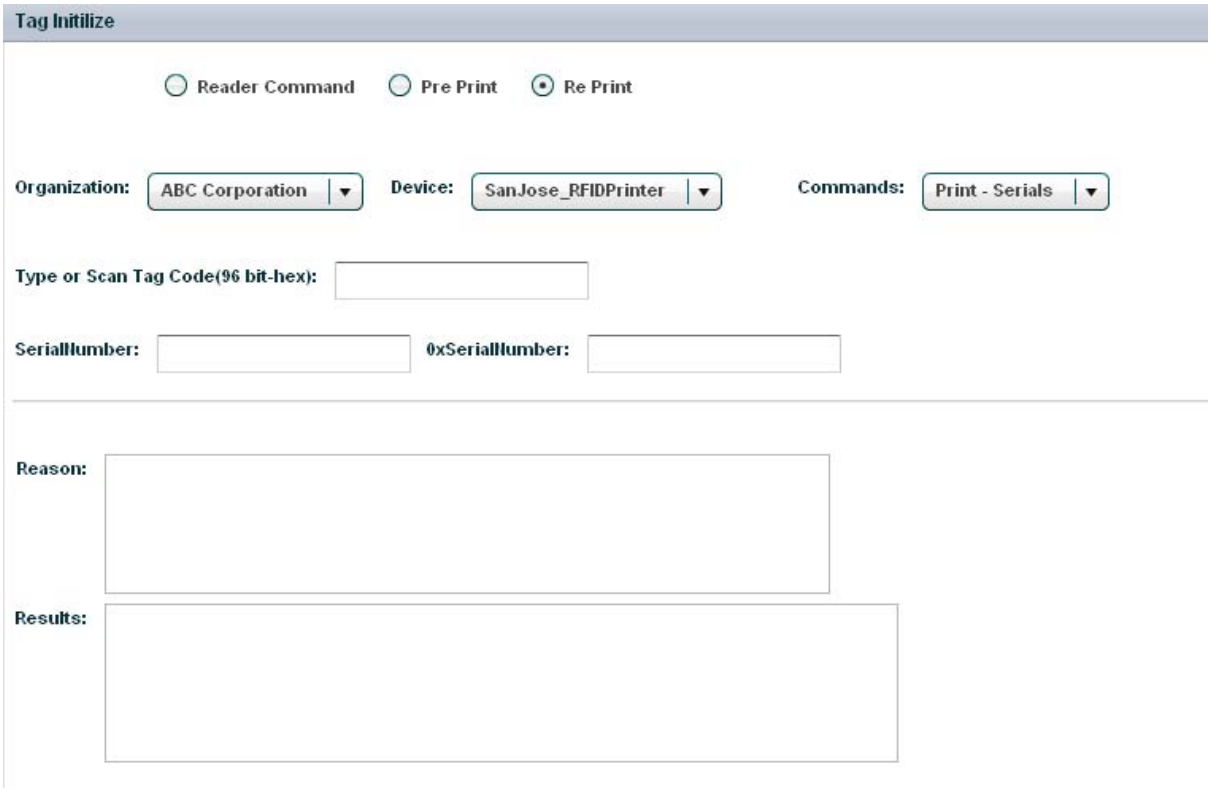
Pre Print is used to Print tag Ids using the RFID Printers. Devices of type “RFID Printer’ should be configured to be able to print.



- Select the Pre Print radio button
- Select the Enterprise organization from the drop down
- The device drop down displays all RFID Printer type devices configured for the Enterprise selected.
- Select the commands from the list of commands.
- Select the encoding scheme and provide the serial numbers.
- Enter the number of tags and the incremental value
- Click ‘Send’

4.20.5 Re Print

The tags printed in the Pre Print can be re printed using Re Print option.



- Select the Re Print radio button
- Select the Enterprise organization from the drop down
- The device drop down displays all RFID Printer type devices configured for the Enterprise selected.
- Select the commands from the list of commands.
- Type the Tag Code or scan the Tag Code to be reprinted.
- Enter the serial numbers and the reason behind the reprint.
- Enter the number of tags and the incremental value
- Click ‘Send’

4.21 GEN2

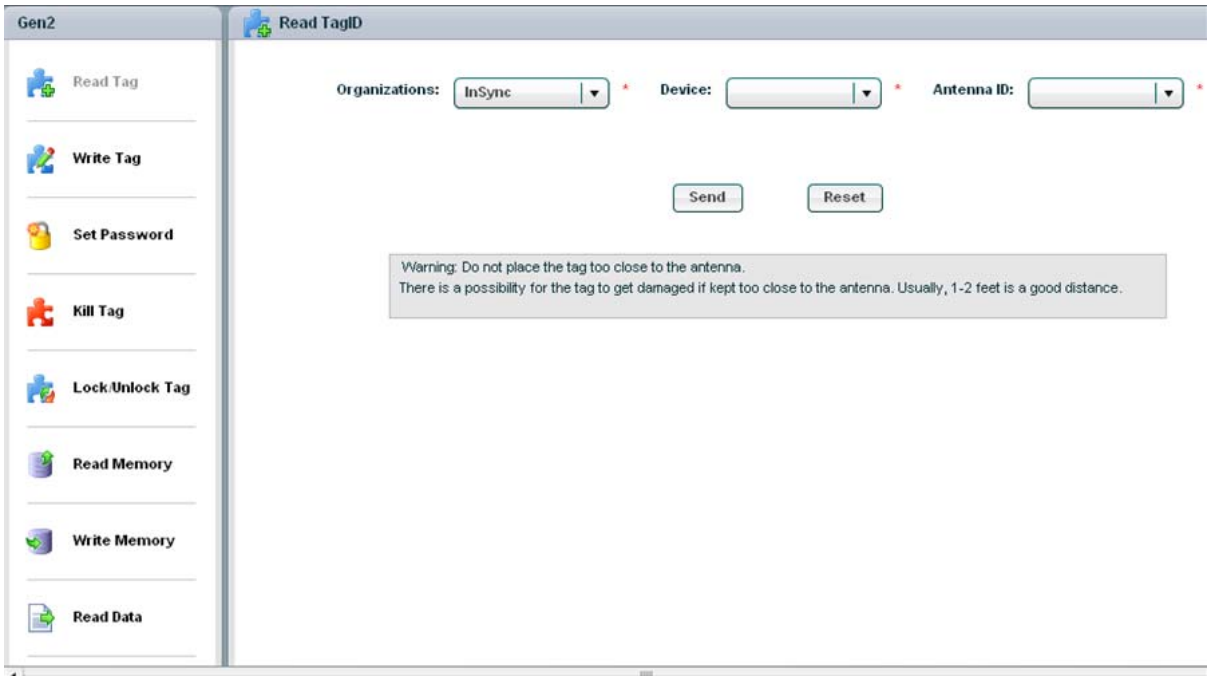
GEN2 is the standard ratified by EPCglobal for the air-interface protocol for the second generation of EPC technologies.

It is designed to work internationally and has other enhancements such as a dense reader mode of operation, which prevents readers from interfering with one another when many are used in close proximity to one another.

Note: For all commands used in GEN2 to work the device should run in MANUAL mode.

4.21.1 Read Tag

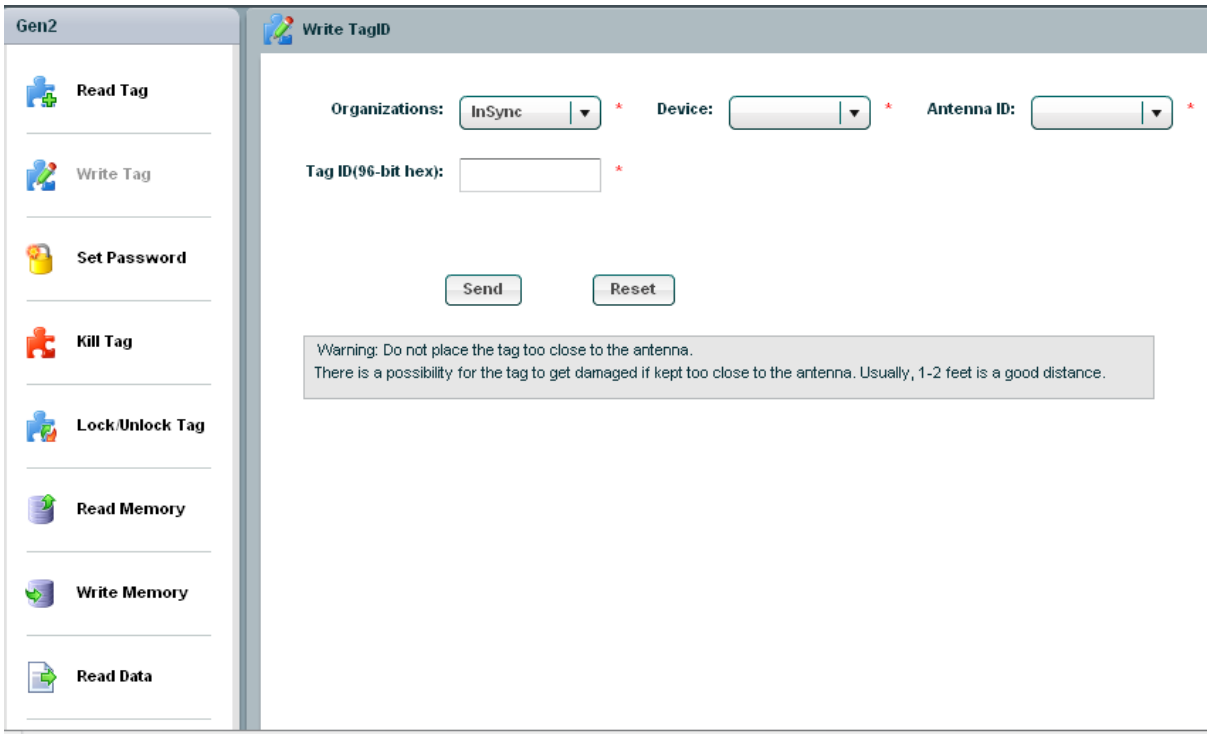
The command is used to read the tag ID of the tags available in a reading zone. Single tag or multiple can be read using this command. Some of the readers can be connected to one or more antennas. A particular Antenna ID can be specified to read the tag(s) or the option ‘ALL’ can be selected to read from any antenna.



- Select the Enterprise organization and the device
- Select the antenna ID if exists for the device
- Place the tag(s) in front of the reader or antenna as the case may be
- Click Send.
- The command will be sent to the reader and will be executed and the results are shown in the Results field.
- If there is any error while executing the command, the same will be displayed in the results. It takes about 4-6 seconds to execute the command and display results here. The actual time depends on the Device settings.

4.21.2 Write Tag

The command is used to write the Tag ID to the tag. For this command to execute successfully, the tag should be writable or write-enabled. If the tag is locked or the tag is READ-ONLY, then the command cannot be executed.



- Select the Enterprise organization and the device
- Select the antenna ID if exists for the device
- Place the tag(s) in front of the reader or antenna as the case may be
- Click Send.
- The command will be executed on ALL the tags that the reader/antenna can read, so be careful to place only those tag(s) that you want to write the Tag ID.

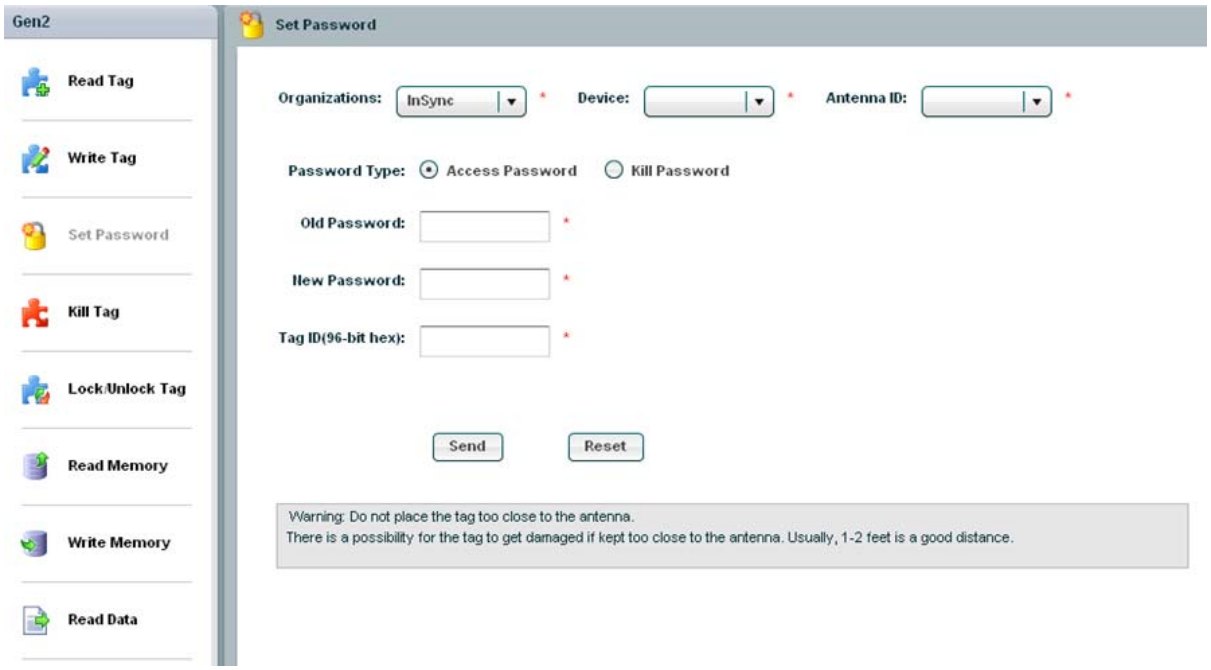
4.21.3 Set Password

Password can be set to access and operate on the tag. This will allow safeguarding the tags and preventing unwarranted access to confidential information.

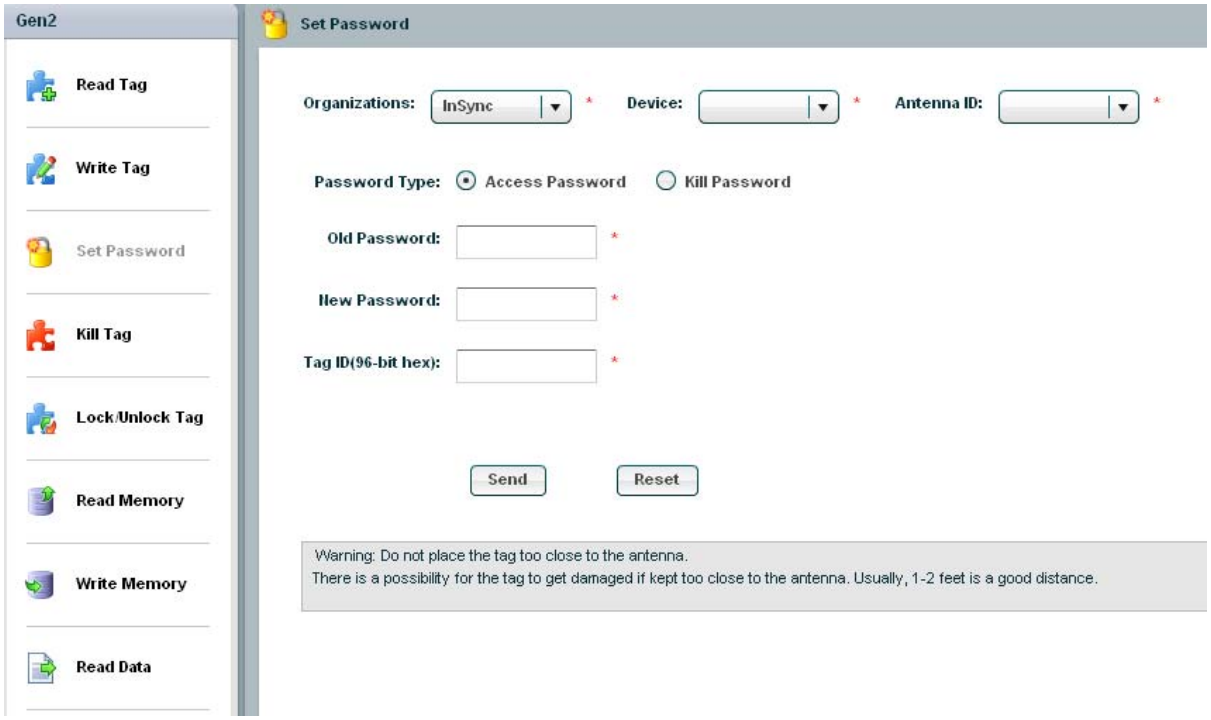
Two types of password can be set

- password to access tag and
- Password to kill tag.

A Tag shall not execute a kill operation if its kill password is zero. A Tag that does not implement a kill password acts as though it had a zero-valued kill password that is permanently read/write locked.

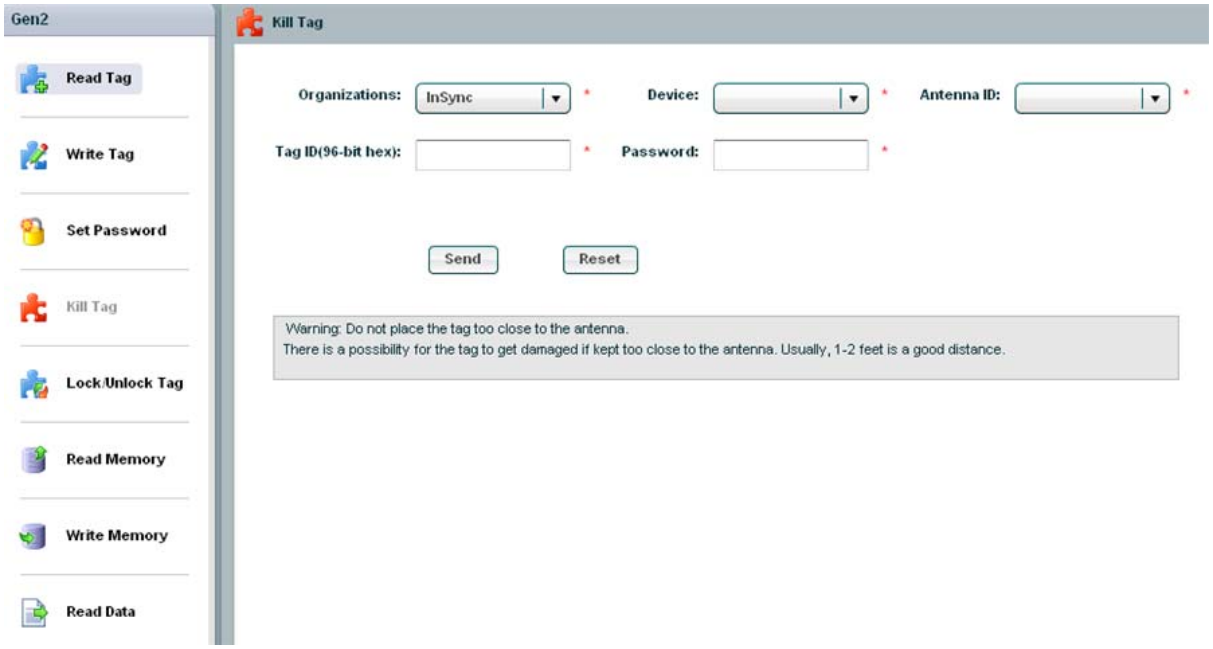


- Select the Organization from the drop down
- Select the device from the device drop down
- Select the Antenna ID
- Check the Access Password / Kill password radio button based on the password that needs to be set.
- Provide the passwords
- Provide the Tag ID to set the password on.
- Click Send.



4.21.4 Kill tag

Tags may be used for temporary purpose or to protect the confidentiality of information. The tags can be destroyed which is called killing of tag. To kill a tag the kill password should be set for the respective tag.



- Select the Organization from the drop down
- Select the device from the device drop down
- Select the Antenna ID
- Provide the Tag ID to be killed
- Provide the kill tag password set for the tag.
- Click Send.

4.21.5 Lock / Unlock tag

4.21.5.1 Lock Tag

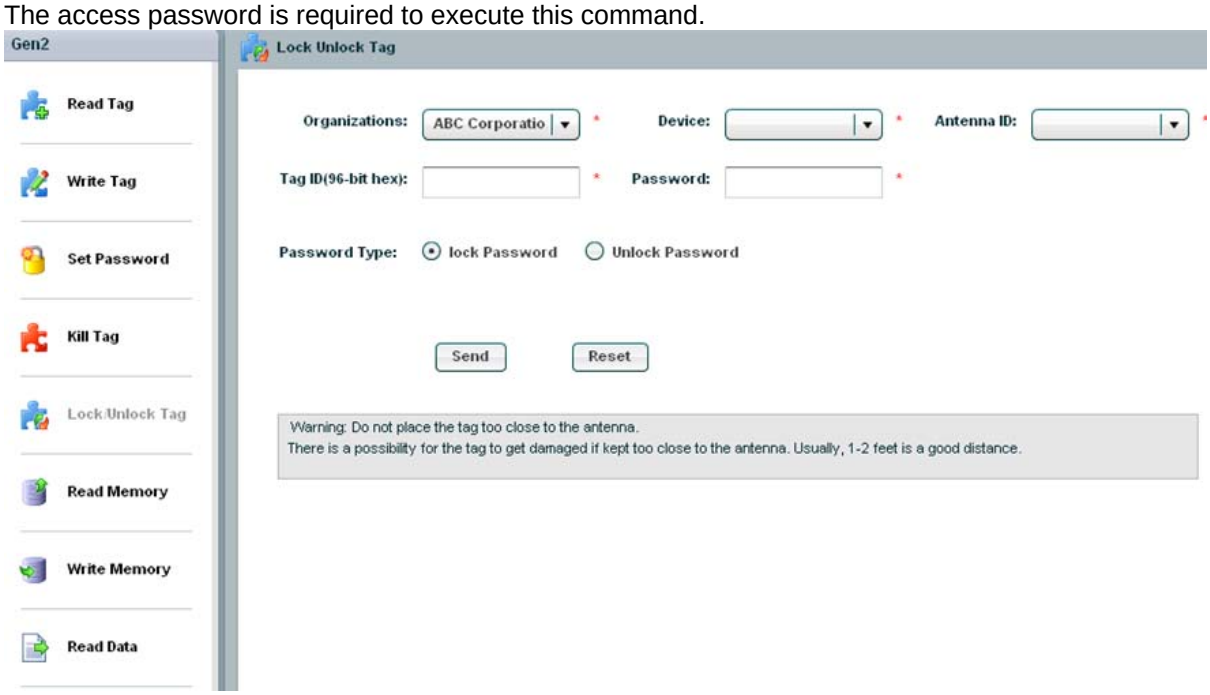
Lock allows an Interrogator to lock the Tag ID field which means further write operations on this field is disabled, but you can read the Tag ID.

The access password is required to execute this command.

4.21.5.2 Unlock Tag

The locked tags can be unlocked using the Unlock Tag function. By unlocking the tag, all operations on the Tag ID field can be done.

The access password is required to execute this command.



- Select the Organization from the drop down
- Select the device from the device drop down
- Select the Antenna ID
- Provide the tag ID to be locked
- Select the ‘lock password’ or ‘Unlock Password’ radio button
- Provide the access password for the Tag ID.
- Click ‘Send’

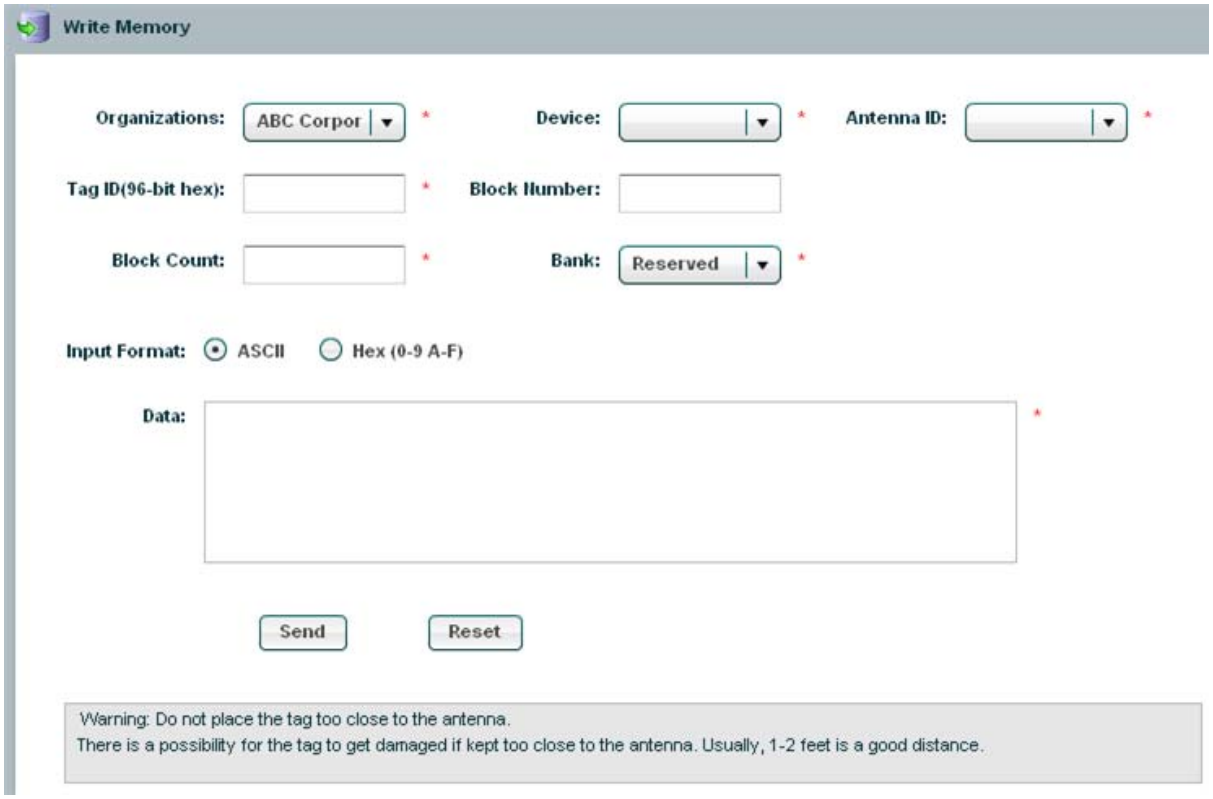
4.21.6 Read memory

The command is used to read the data of a tag from any Memory Bank starting at any Block Number till any Block Count.

- Select the Organization from the drop down
- Select the device from the device drop down
- Select the Antenna ID
- Provide the tag ID from where the data should be read
- Set the block number and the block count
- Select the memory bank where the data has to be read from
- Select the Result format to view the results in the Data area.
- Click ‘Send’

4.21.7 Write Memory

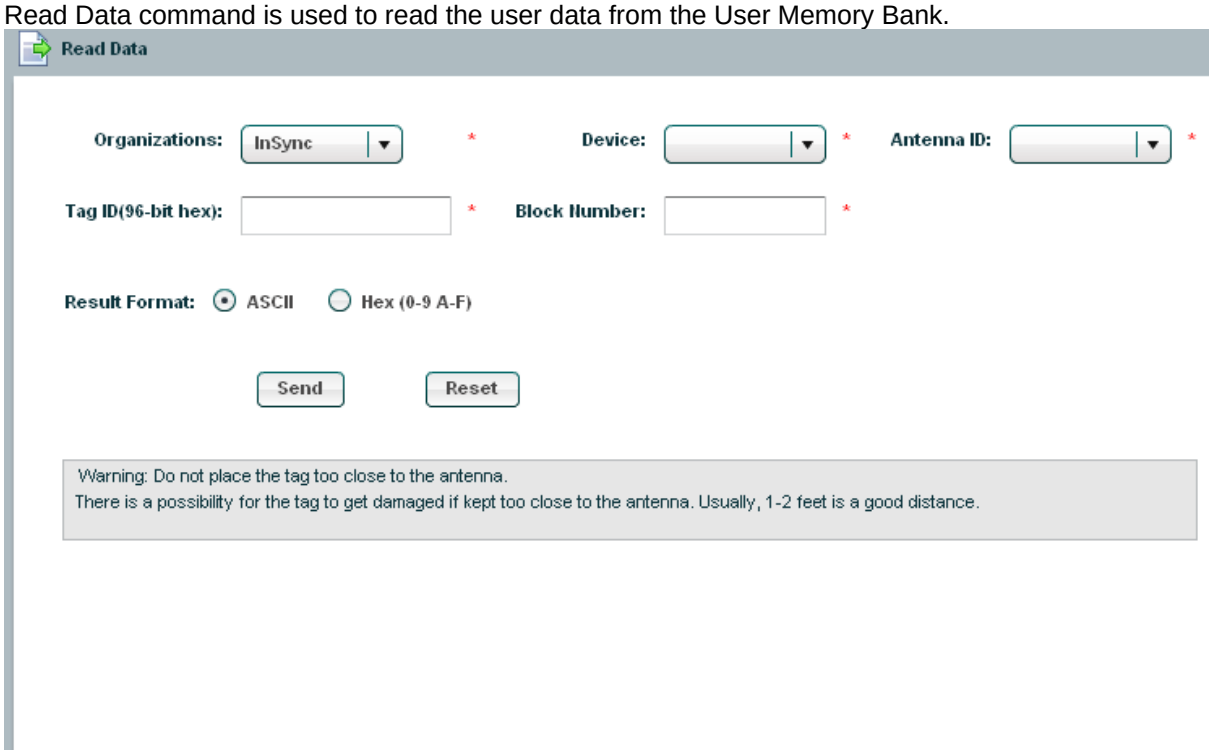
The Write Memory command is used to write any Data in any Memory Bank starting at any Block Number till any Block Count.



- Select the Organization from the drop down
- Select the Device and Antenna ID.
- Enter Tag ID of the tag on which data has to be written.
- Enter the Block Number of the block on which data has to be written.
- Enter the Block Count.
- Select the memory bank
- Select the format in which data should be entered into the Tag Memory.
- Enter the data to be written on the tag memory.
- Click **Send**.

4.21.8 Read Data

Read Data command is used to read the user data from the User Memory Bank.

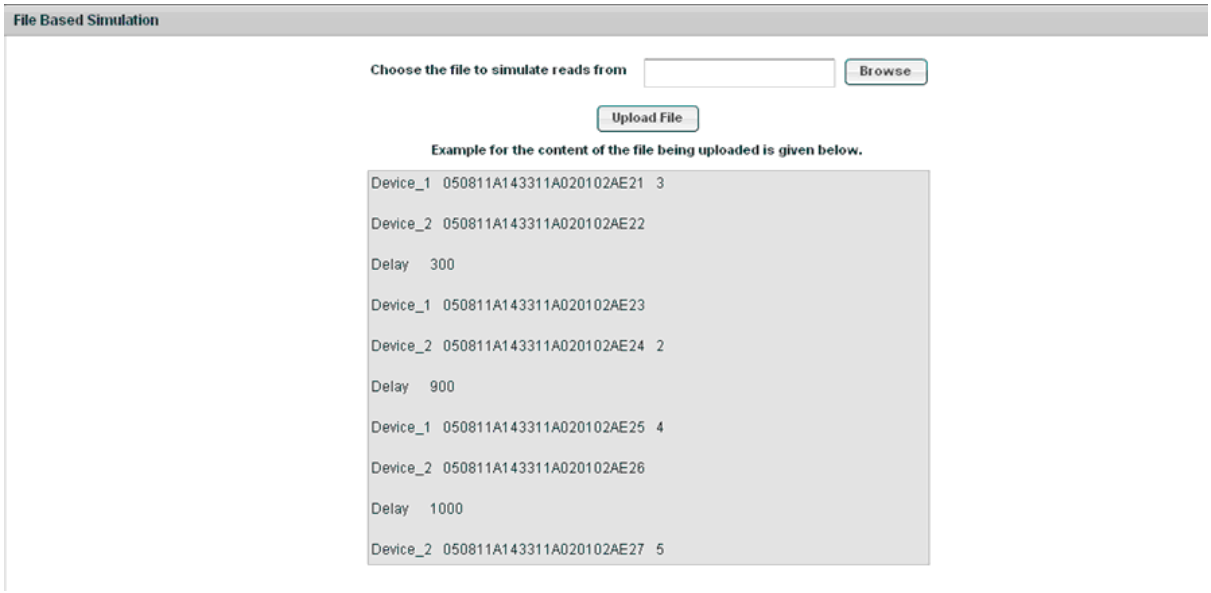


- Select the Organization from the drop down
- Select the Device and Antenna ID
- Enter Tag ID of the tag from which data has to be read.
- Enter the Block Number of the block to be read.
- Select the format in which to view the results.
- Click **Send**.

4.22 File Based Simulation

File based simulation is similar to Simulate Read where simulation of the reads happens to replicate a scenario of reading a tag using a device. The tags to be simulated are stored in the file which can be provided as input to the Simulator.

This helps to reuse the simulate codes and also makes it easy to simulate reads any number of times.



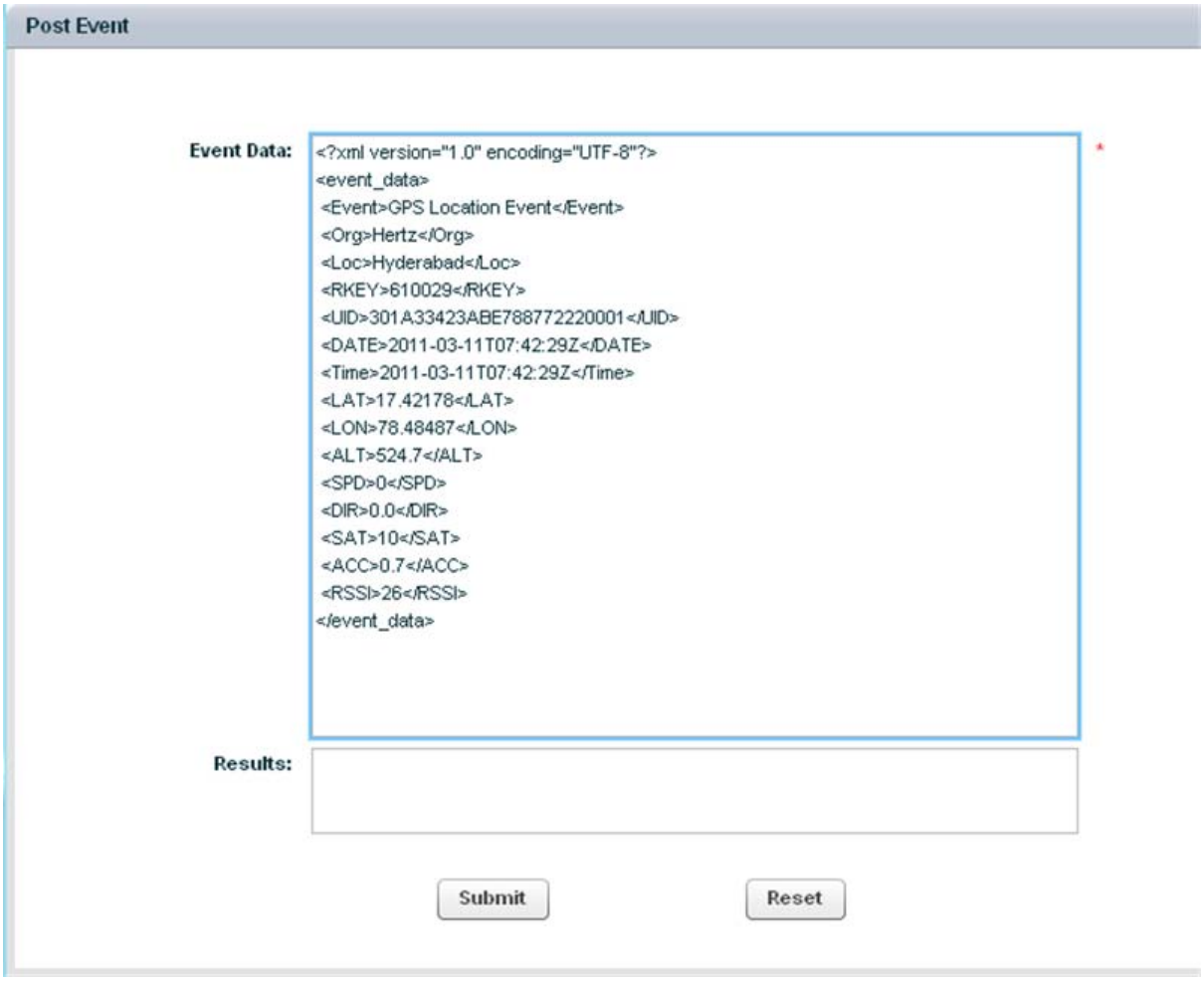
- Browse to the file having the tags
- Click Upload file

Note: Use the example provided on the page to format the file with the tags.

- The first parameter is the Device Name
- The second parameter is the tagID
- The third parameter is the repeat count which is the no of times the same tag gets simulated. The default value is 1.
- Delay keyword makes the simulation sleep for the interval defined.
- Each of the parameter is tab or space separated.

4.23 Post Event

The post event page is used to simulate an event in iApp for further processing by the handlers or other tasks that the event can be acted upon as configured for the event.



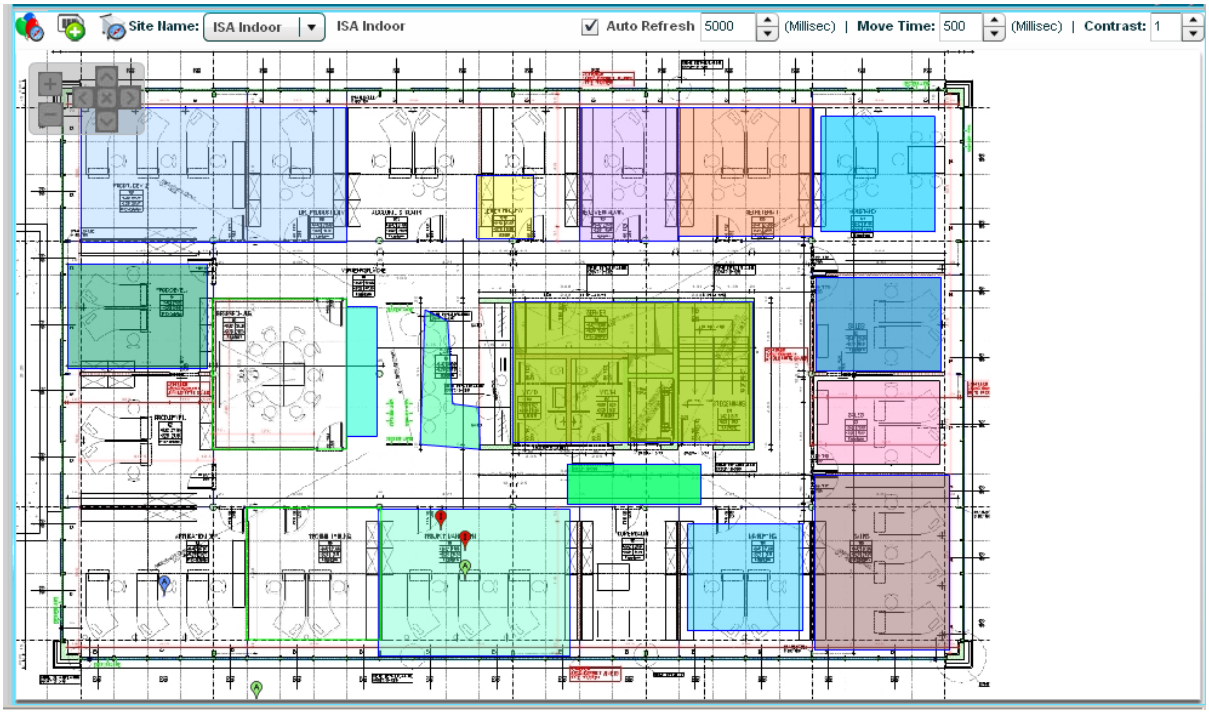
Attributes	Description
Event Data	Enter the event xml in the structure defined in iApp.
Result	Success / Failure based on whether the event gets posted to iApp or not.

Note: The post event simulates as if the event is raised in iApp. Processing of the event is based on how the event has been set in Event management. iApp logs all exceptions to the log file if it is not successful in processing the event.

4.24 RTLS

RTLS page gives a snapshot of the plant view and the inventory at the RTLS site configured in the system. The location should be configured as an RTLS site while configuring trhe Location hierarchy and the image for the location should be calibrated.

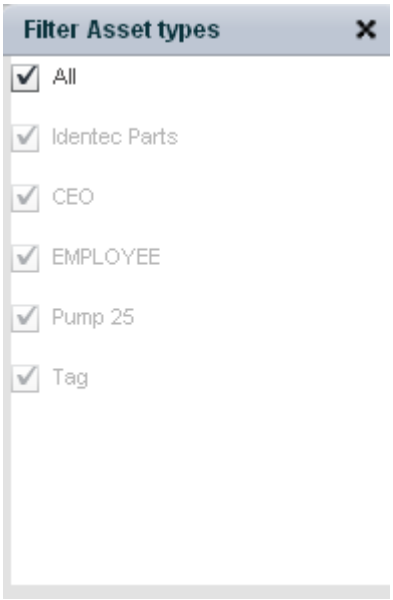
4.24.1 Plant View



The viewer has all the assets plotted at the selected RTLS location.

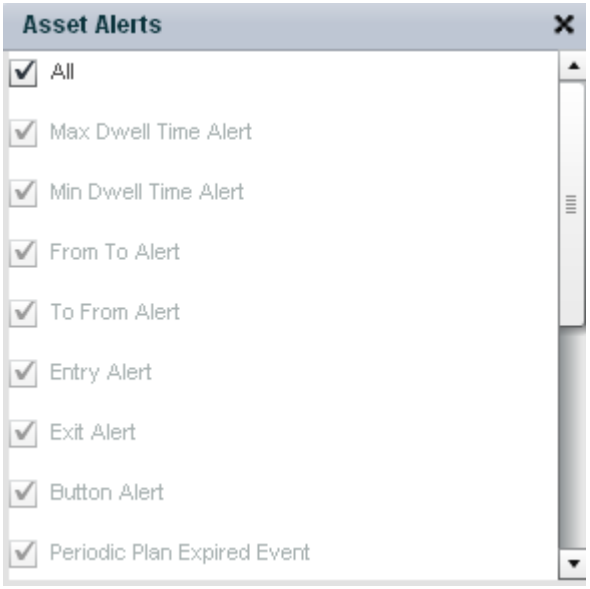
The viewer has the following options

- Show Asset Types – Lists all asset types in the system which can be used to filter the assets on the viewer.

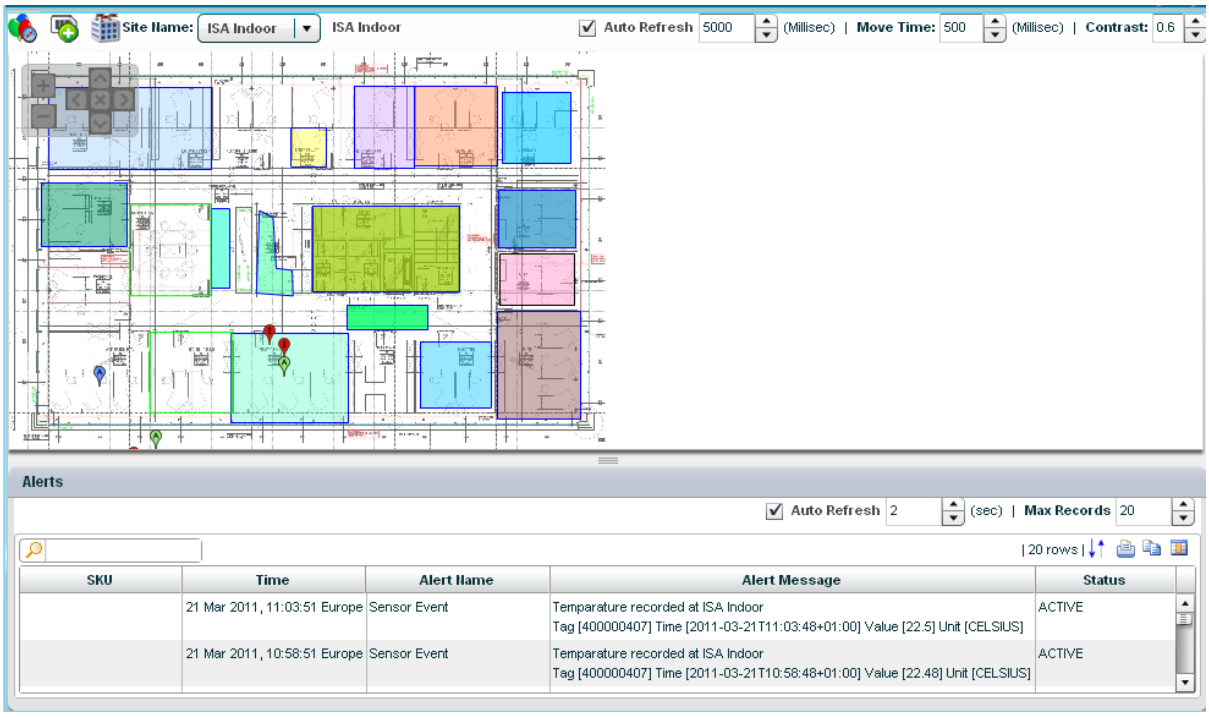


The asset types can be selected/deselected based on which the assets on the viewer get filtered.

- - Asset Alerts : shows a listing of all asset alerts that will be visible in the alerts panel below the viewer by clicking on Show/Hide Alerts. Select/deselect the alerts from the panel based on which the alerts will be displayed in the panel.



- - Show / Hide Alerts: Click to view the alerts panel with the listing of all alerts at the RTLS location selected.

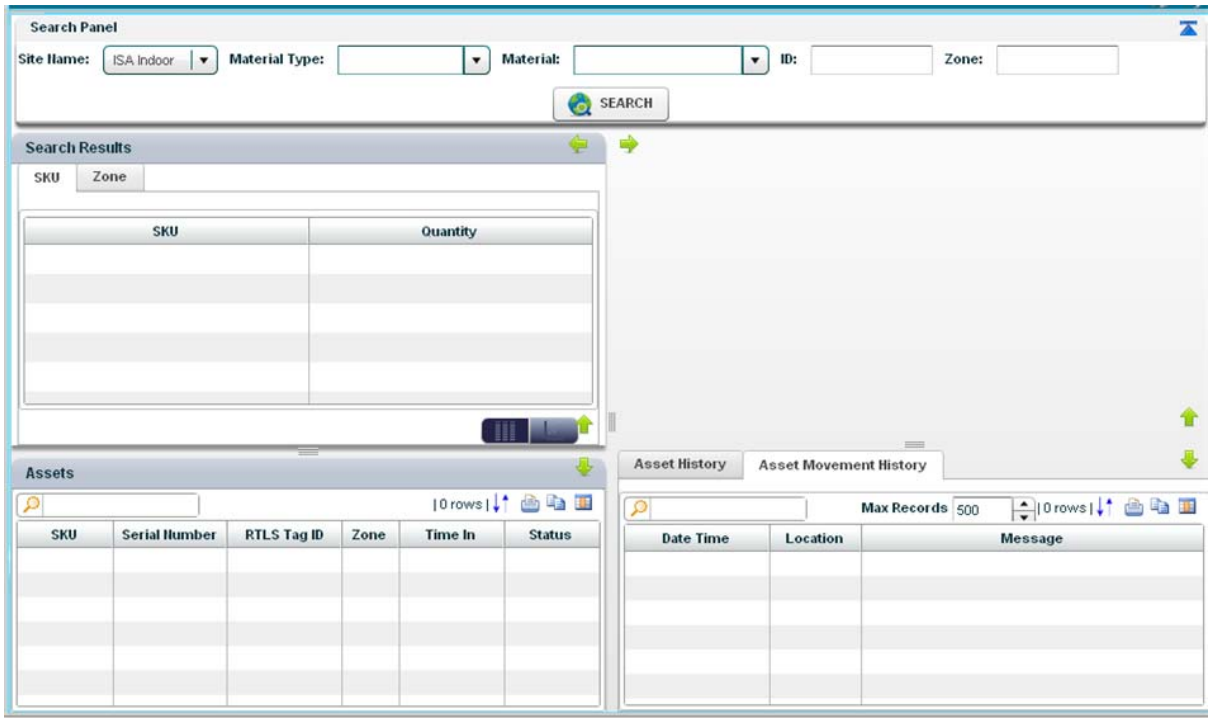


- Auto Refresh – The viewer can be configured to auto refresh every defined interval

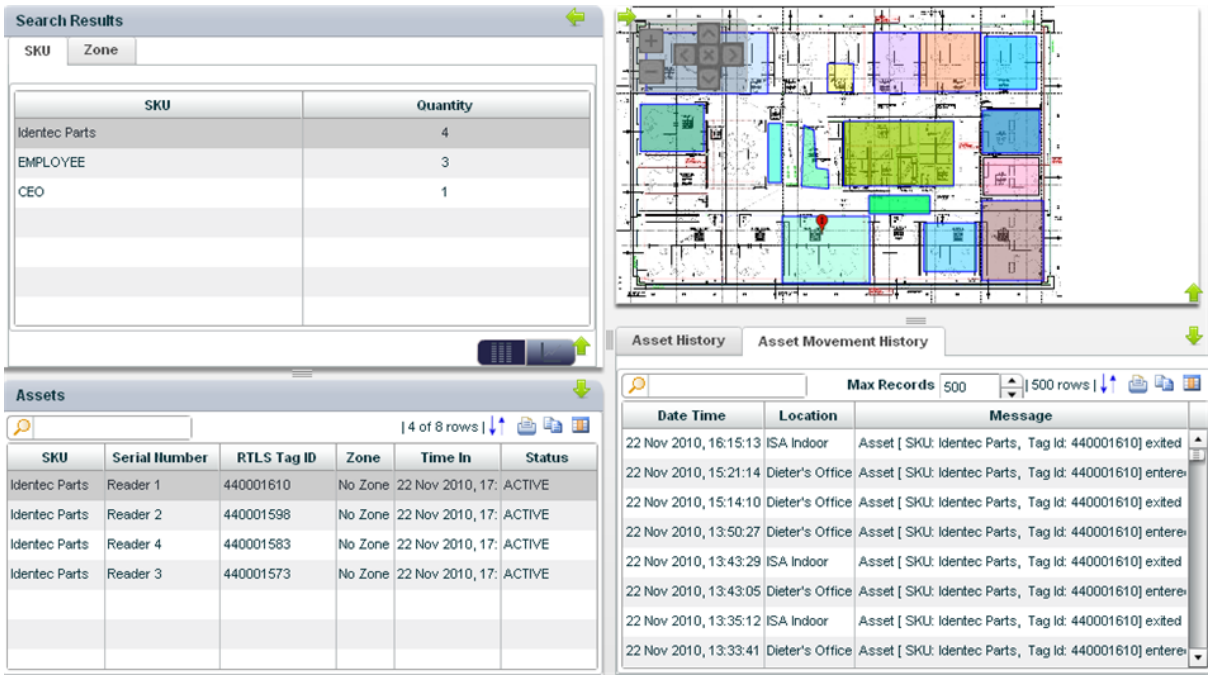
- Move Time – is the time the viewer should take for the asset to plot from one location to another location when there is a change in its physical co-ordinates. The move time should be adjusted so that there is a smooth movement of the asset on the viewer when its co-ordinates change.
- Contrast – Adjust the contrast for the image.

4.24.2 Inventory

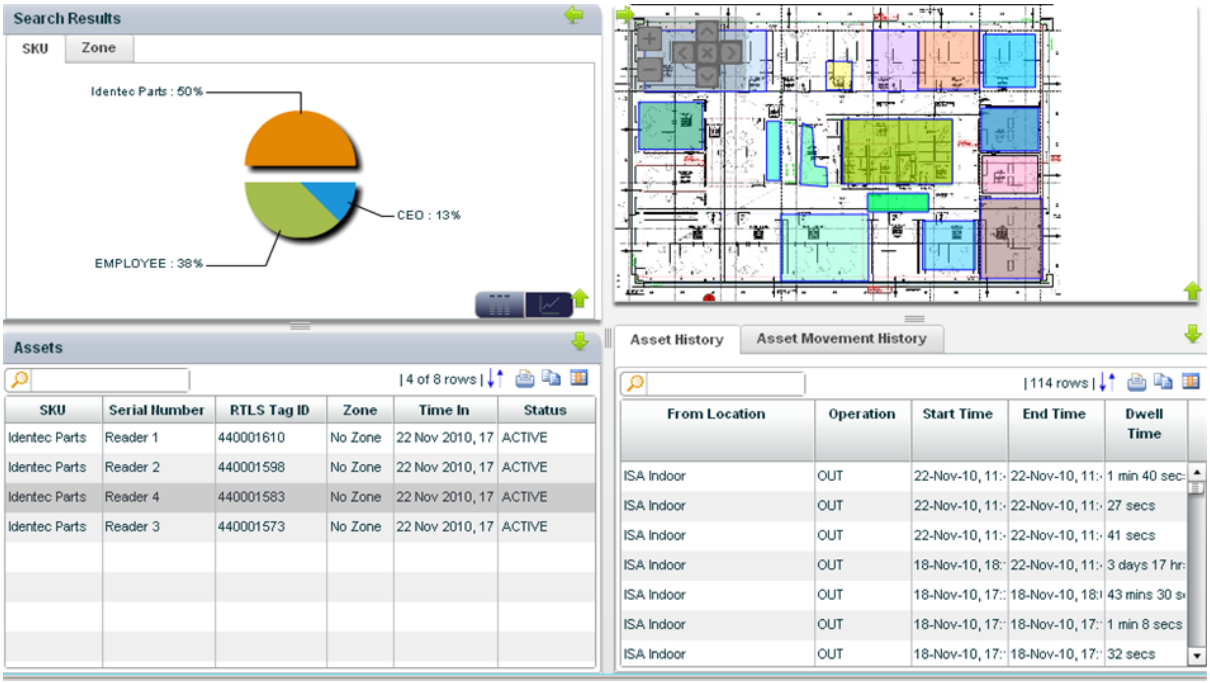
The inventory shows the inventory at the given RTLS site**.** The search panel gives the options to filer by RTLS site, Material Type, Material, ID and zone (location region). The ID and the zone are what are configured at the time of creating the location as RTLS site.



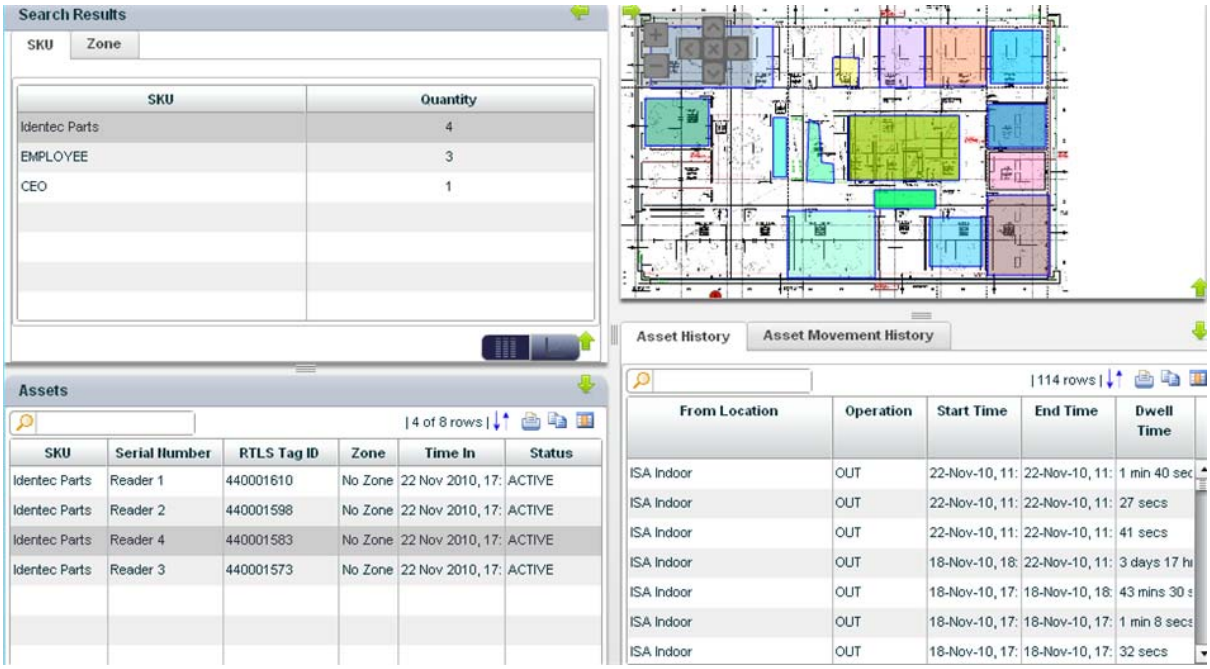
- The inventory is shown in a 4 panel view with each view giving drill down of the results.
- Select the search options and click on Search.



- The search results are shown by SKU and Zone in the top left panel. Both the SKU and the zone panels can be viewed as a grid or a pie chart based on . Clicking on shows the chart view.



- The top right panel shows the inventory on the calibrated image. Selecting the row in the SKU/Zone grid filters the assets in the right side image viewer along with the botton left assets panel grid and the botton right side History panels.
- Selecting the asset in the assets panel refreshes the top right viewer and the bottom right History panels.



- Each of the panels can be resized using the , , , buttons on the panels.

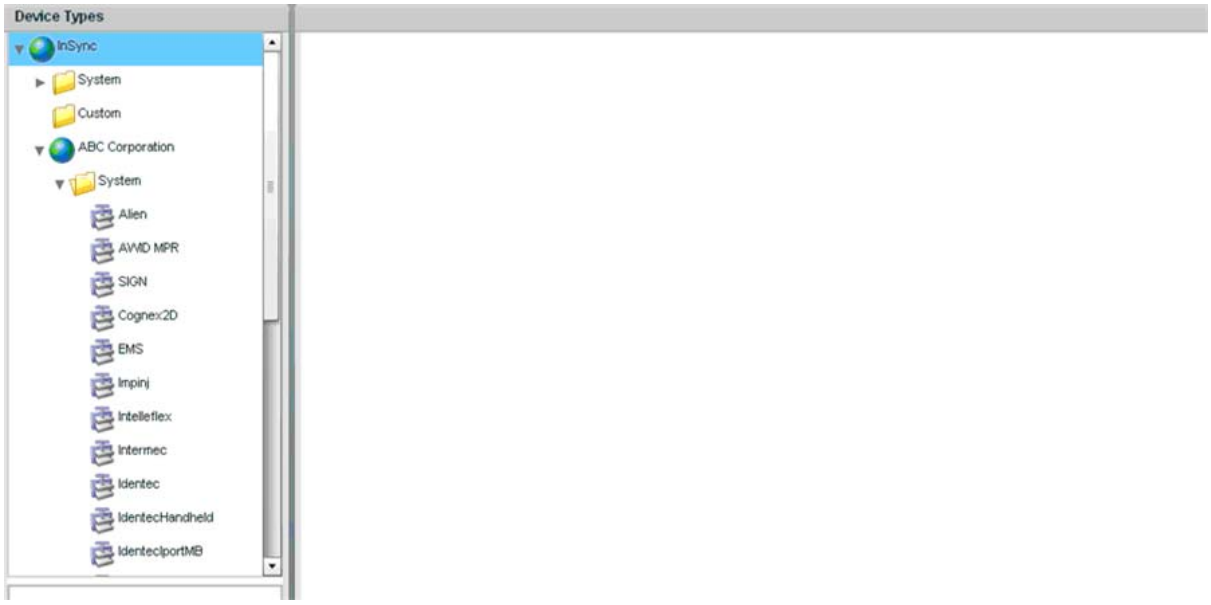
4.25 Software

The Software tab helps in loading and maintaining the implementation information of the various meta data entities used in the Configuration and EMS section. This includes

- Device Types
- Event Handlers
- Jobs
- Connections
- Receiver Types

4.25.1 Device Types

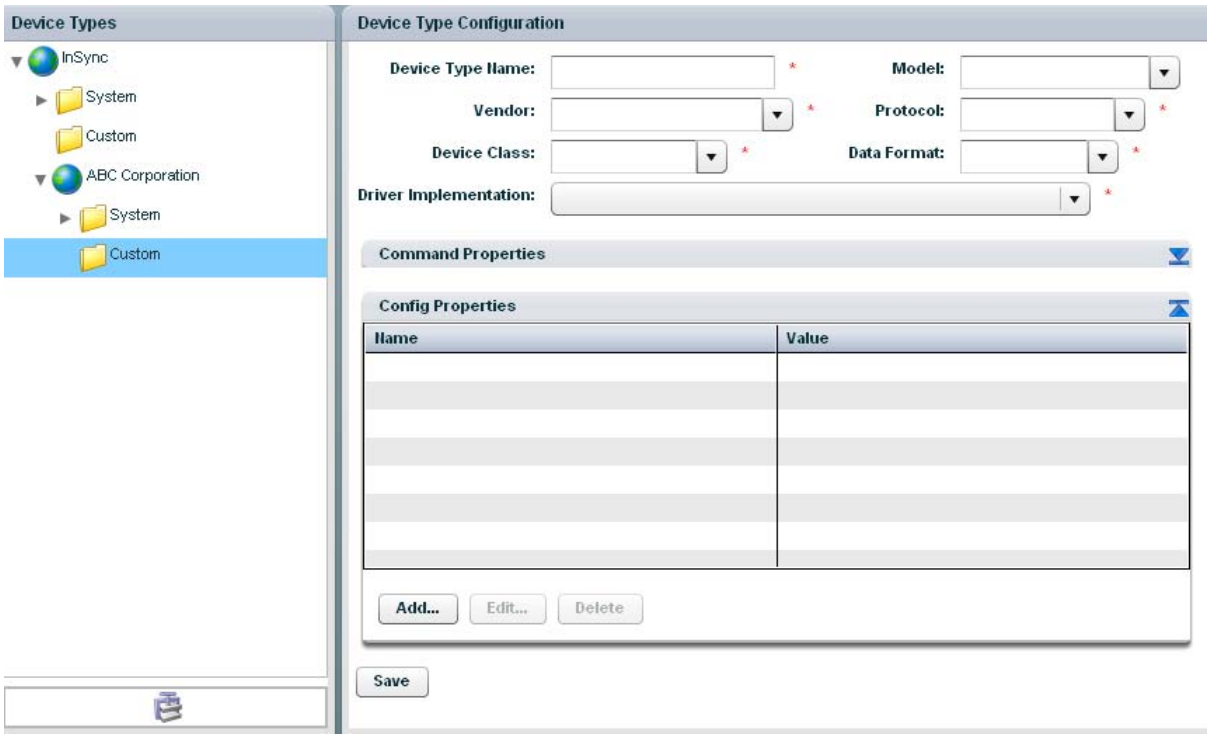
Device Types refers to RFID readers which are categorized into various types based on the communication protocol, command set supported. IApp provides different Device Types so that Devices can be defined for each Device Type.



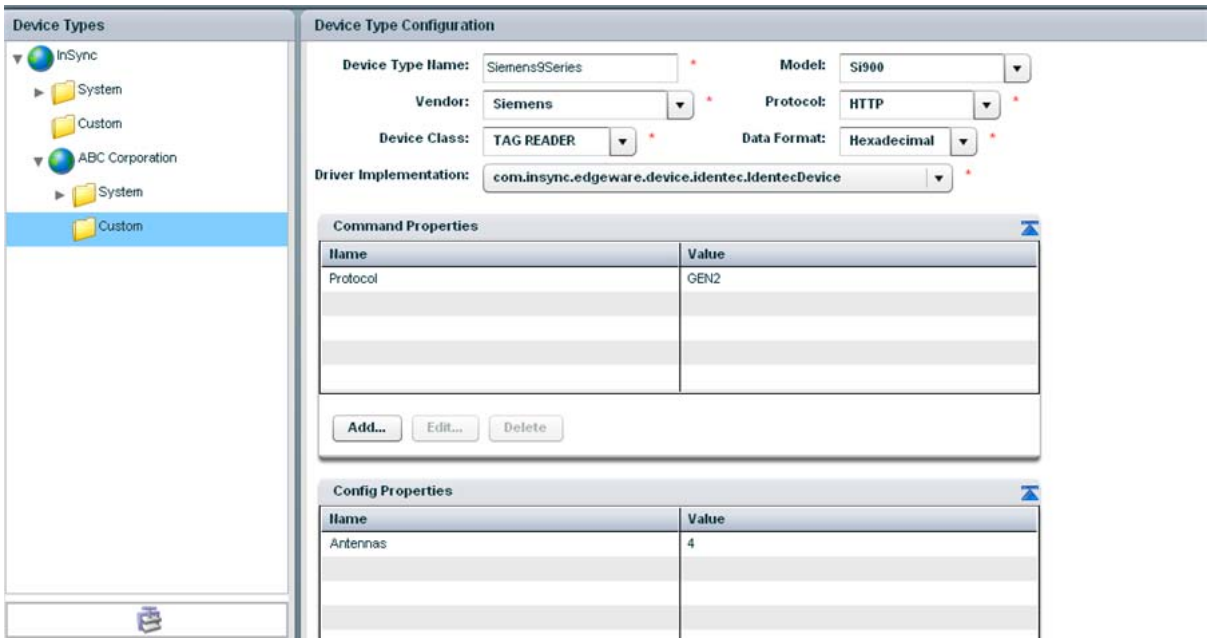
- The Device Types are listed by System / Custom.
- There are a set of System Device Types provided by IApp that can be used while configuring the devices.
- The System Device Types can be added at the Home Enterprise only
- Custom device types can be added at each Enterprise Organization

4.25.1.1 Create Device Type

- Select Enterprise Organization
- Select System / Custom
- Click



Attribute	Description
Device Type Name	Name of the Device Type
Model	Model name
Vendor	Vendor Name
Protocol	Communication protocol supported by the device type
Device Class	Represents the type of device.
Data Format	Format of the data the device understands
Driver Implementation	Java implementation of the device driver
Command Properties	The properties supported by the driver implementation and are variables built into the implementation and are in the form of name value pairs. The property values can be static values as well as dynamic. The dynamic variables are passed to the driver while configuring the device. The Command properties vary for different device types
Config Properties	Are used to configure and fine tune the device to run properly and which are understood by that device type only. These properties vary for different device types



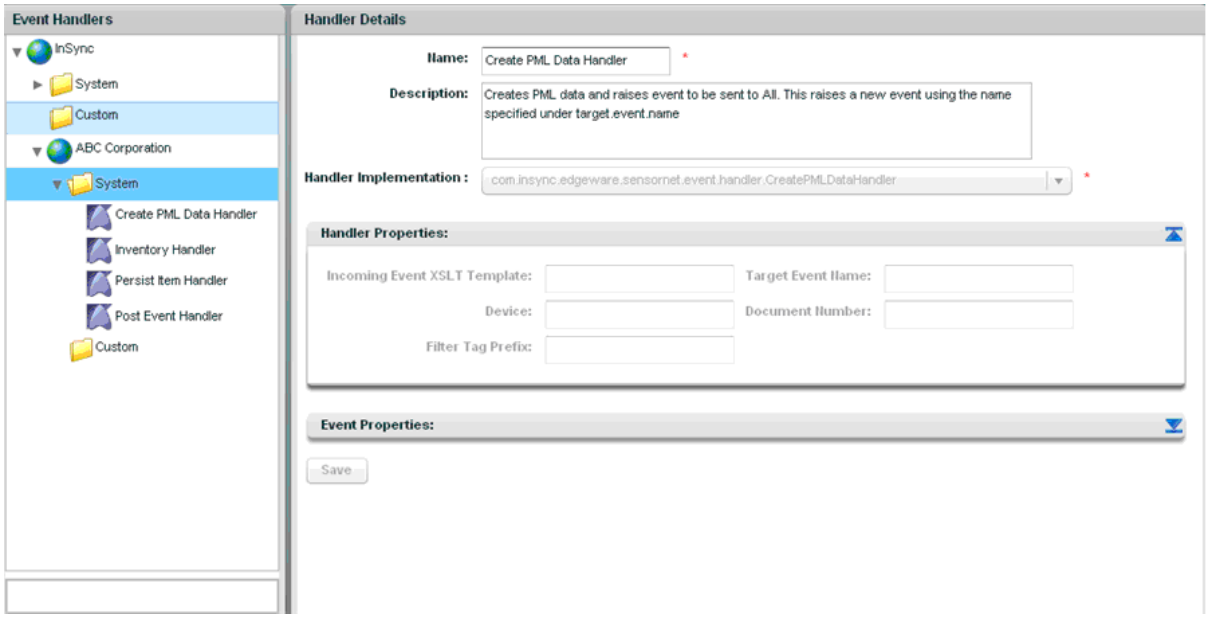
- Click ‘Save’ to save the Device Type to the tree
- Devices of the Types created here can be created or configured from the Configuration->Explorer or the Configuration->Devices section.

Note: Refer to IApp Device Types section for more information on the different device types existing the IApp.

4.25.2 Event Handlers

Event Handlers are specific implementation of a task which acts on the Event to take appropriate actions based on the implementation. Event handlers are configured for Events while defining the Actions for the Event.

Each Event handler is a java implementation which understands a set of properties (Handler Properties) both internally and as well as on the Event (Event Properties) which it will act on if configured for that Event.

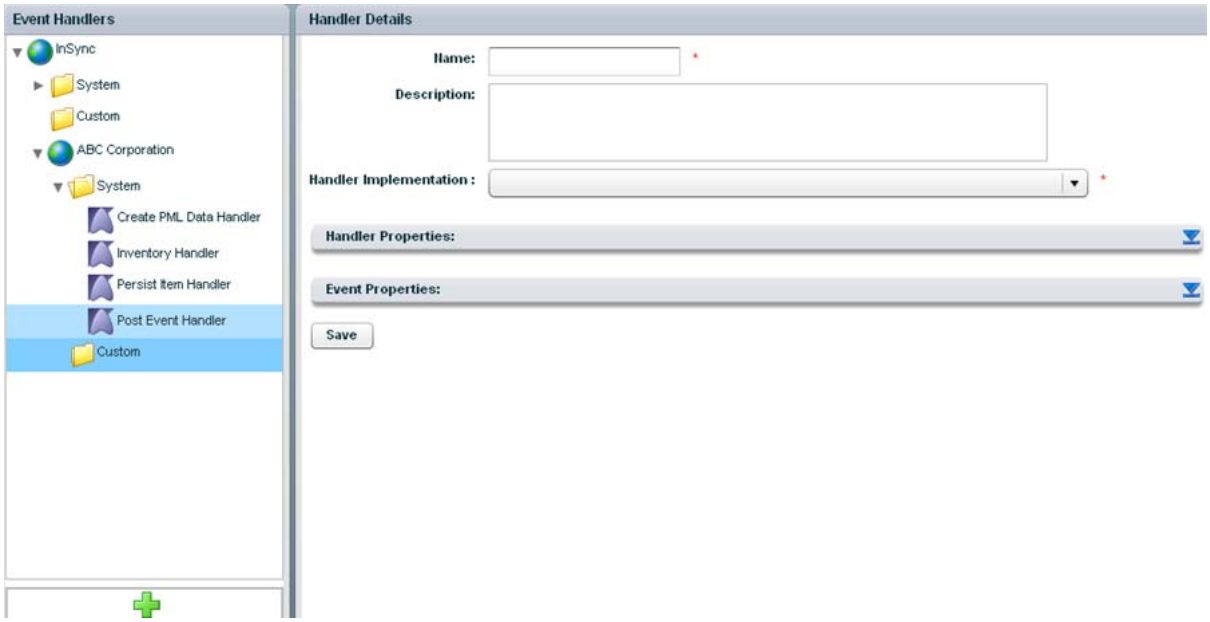


- Event Handlers are listed by System / Custom.
- There are a set of System Event Handlers provided by IApp that can be used while configuring the actions for the Event.
- The System Event Handlers can be added at the Home Enterprise only
- Custom handlers can be added at each Enterprise Organization

Note: For more information on how a Event Handler is designed refer to IApp EventHandler Design section.

4.25.2.1 Create Event Handler

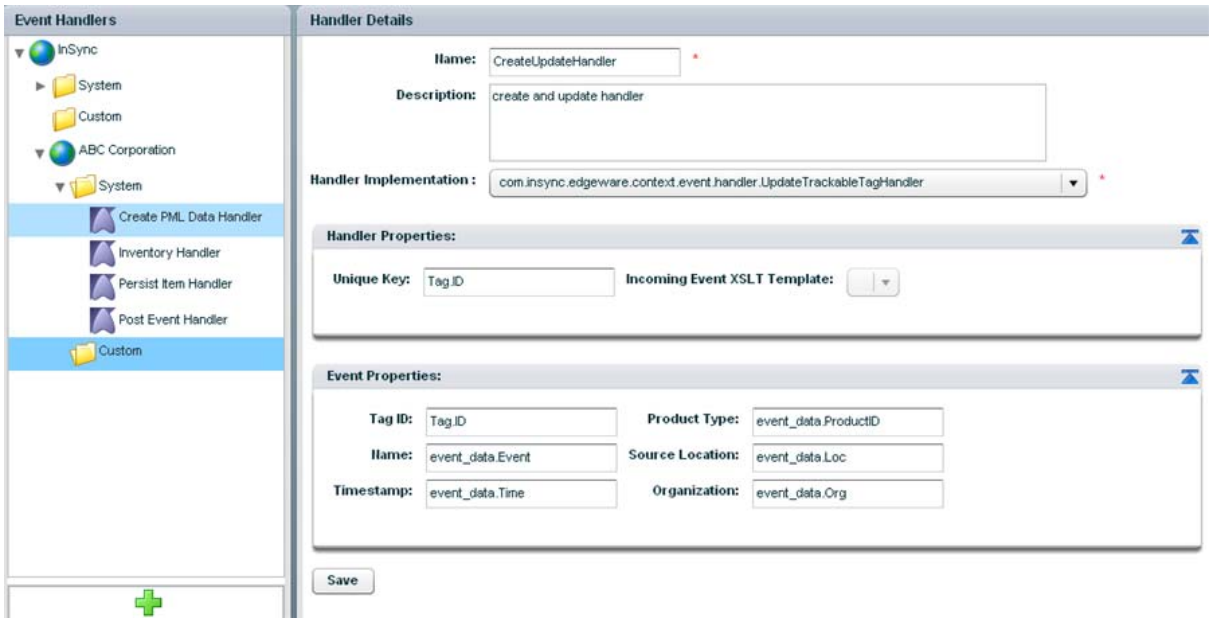
- Select Enterprise Organization
- Select System / Custom
- Click



Attribute	Description
Name	Event Handler name
Description	Short Description
Handler Implementation	Java implementation of the handler. Select the Handler from the drop down.
Handler Properties	Properties which are built into the handler implementation. The property can be static values or dynamic values which are set while defining the actions for the event. The Handler properties vary for different Handler implementations.

Event Properties	Tells the handler the structure of the Event it has to act upon. Each property built into the Handler and which is understood by the handler is mapped to the Event element.

Note: The Handler Properties and the Event Properties are decided by the Handler Implementation.



- Click ‘Save’
- The created handler gets added to the handler tree in the left panel.

Note: The Handler properties and Event Properties cannot be edited from this page. They can be modified at the time of creating the Actions on the Event.

4.25.2.2 Export / Import

Event Handlers can be exported at each Enterprise Organization and imported to a different system

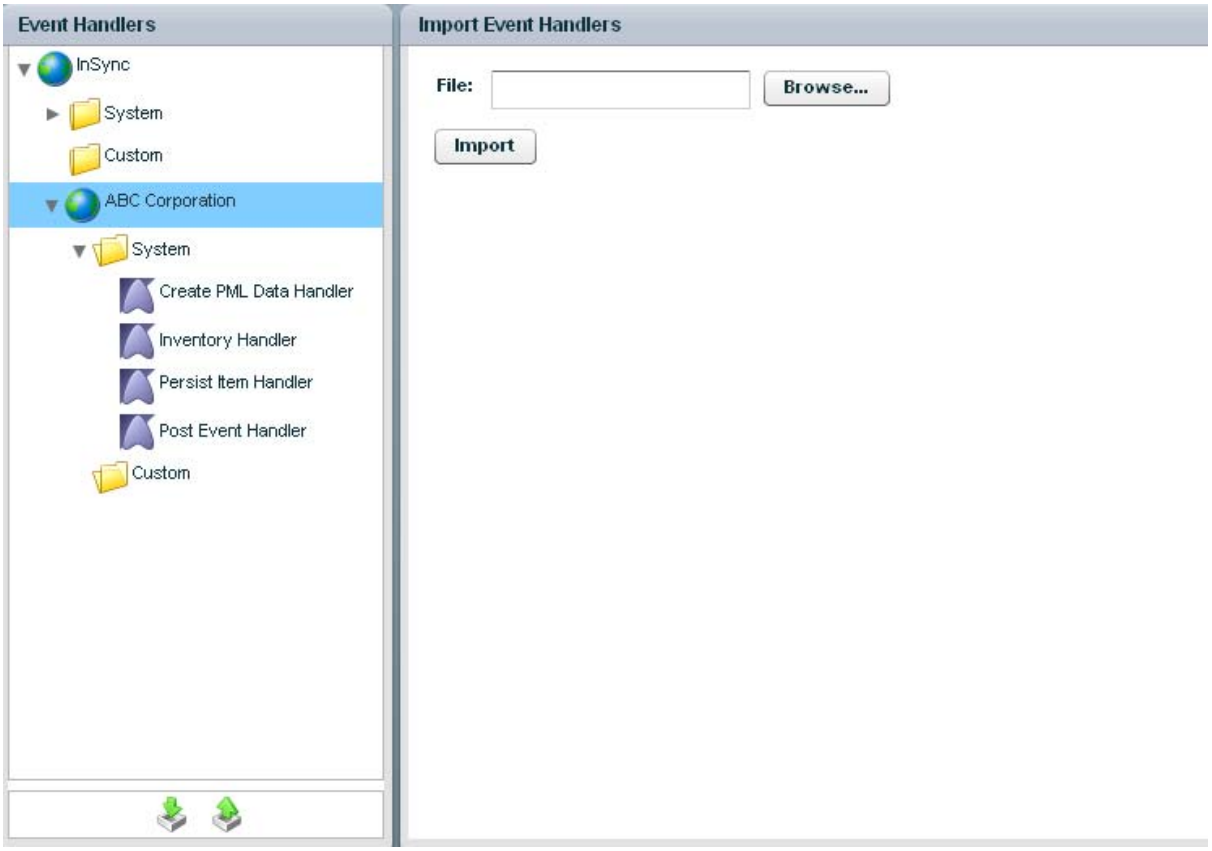
- Export at the Home Enterprise exports the Handlers at all the Enterprise Organizations existing within the Home Enterprise
- Export at Enterprise Organization within the Home Enterprise exports Event handlers at that enterprise only.

Export

- Select the Enterprise Organization and click on
- Select the location to save the file and click Save
- EventHandler.xml gets exported to the selected location.

Import

- Click in the left side lower panel

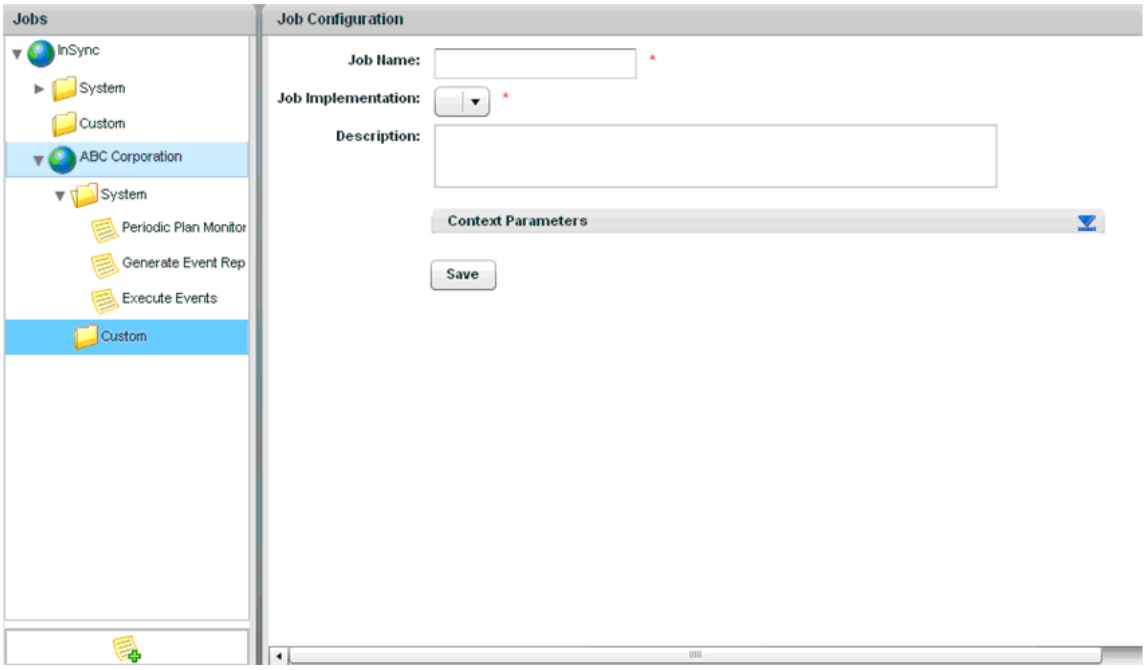


- Browse to the folder where the xml file to be imported is located
- Click Import button

Note: It is required that the Enterprise organization for which the handler import is happening and which exists in the .xml file exists in the system before the import happens.

4.25.3 Jobs

Job is a single logical unit application consisting of several steps. Jobs in IApp can be created based on the implementation and can be scheduled from the Scheduled Jobs page of Event Management Service.



- Jobs in IApp are created as System or Custom and are listed by Enterprise Organization.
- IApp provides two system jobs by default.
- The System Jobs can be added at the Home Enterprise only
- Custom Jobs can be added at each Enterprise Organization
- Only one job is allowed to be created for each Job Implementation

4.25.3.1 Create Job

- Select the Enterprise Organization
- Select system or custom and click on

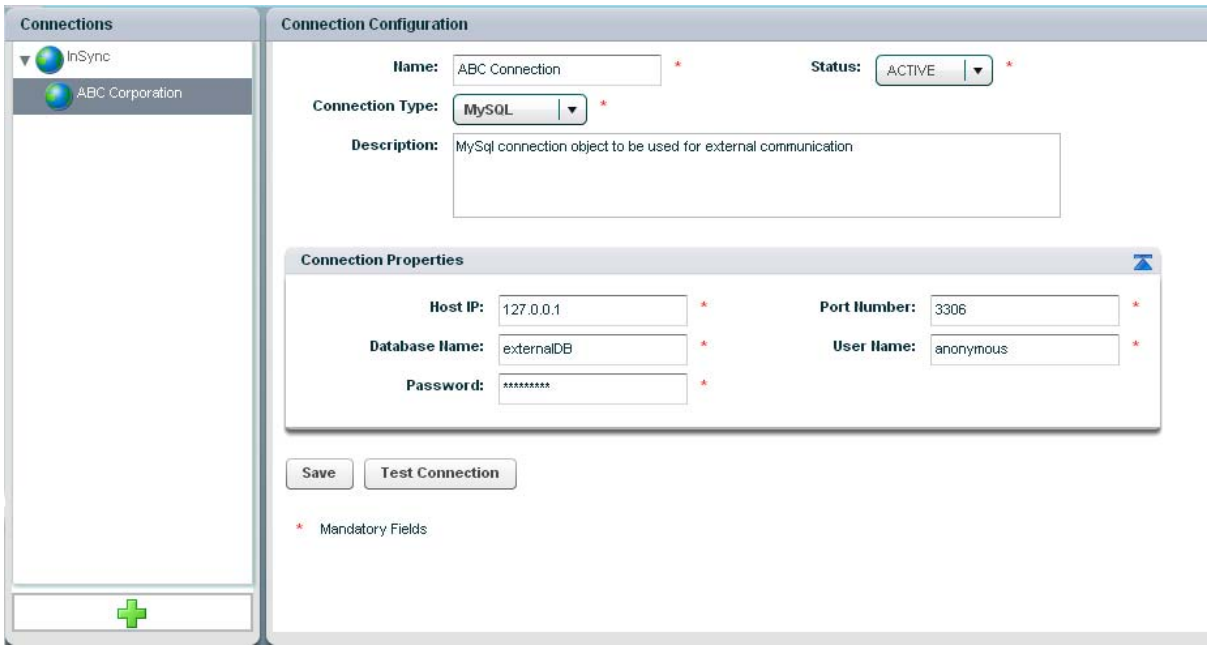
Attribute	Description
Job Name	Name of the job
Status	Active / InActive
Job Implementation	Java implementation of the job
Description	Short description
Context Parameters	These are properties understood by the Job implementation and which are required for the functioning of the job. The parameters are name value pairs for which the values can be provided at the time of Scheduling the job.

Note: The Context Parameters can be edited at the time of Scheduling the Job from the Schedule Jobs page of Event Management Service

4.25.4 Connections

For applications trying to communicate to external systems, IApp provides the infrastructure to connect to external systems using the connections created.

IApp currently provides MySql/Sql Server/Oracle connection types based on which connections can be created.



- Connections are provided for each Enterprise Organization
- Select the Enterprise and click on to create a connection

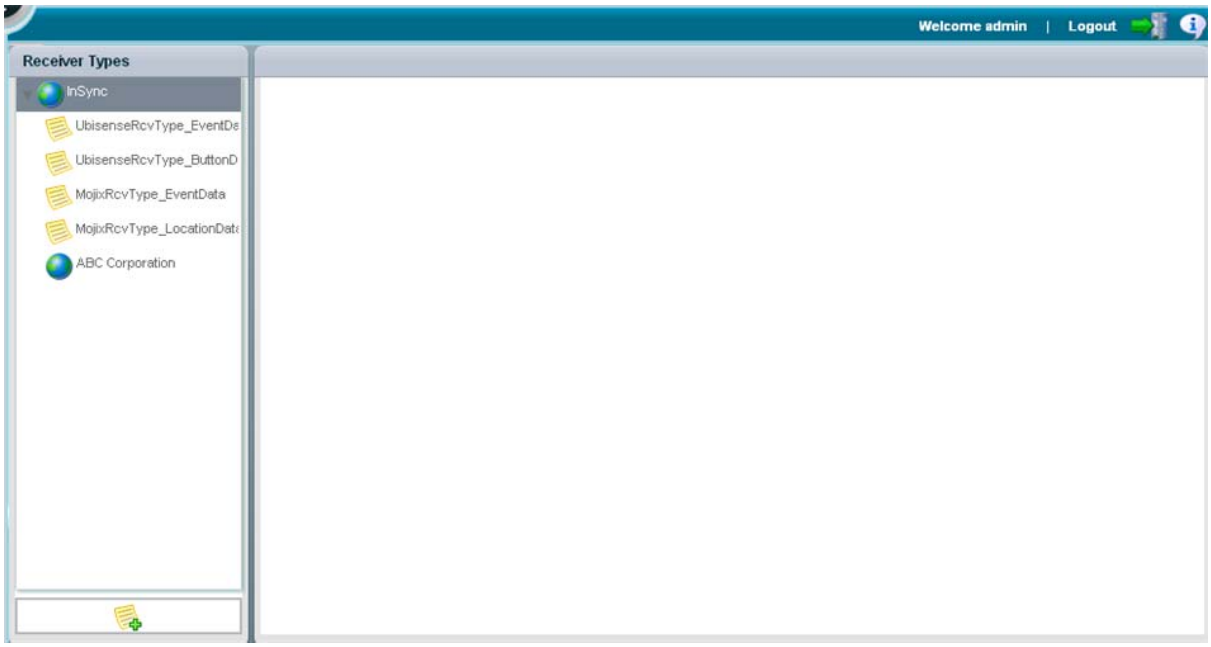
Attribute	Description
Name	Unique name for the connection within each Enterprise. Applications use this connection name to get a connection object and operate on the connection.
Status	Active / InActive
Connection Type	The supported types are MySql / Oracle / Sql Server/DB2/email. Select the type from the drop down. Based on the drop down the connection properties vary.
Description	Short description
Connection Properties	Based on the connection type selected the properties panel change. These are connection properties which are necessary to get a connection from the selected type.

Note: Consult the network IT administrator for the connection parameters while creating the connection.

4.25.5 Receiver Types

Receiver types are specific implementation of a receiver which acts on the incoming event structure to take appropriate actions based on the implementation. Receiver types are used while creating the receivers in the Event Management->Receivers panel.

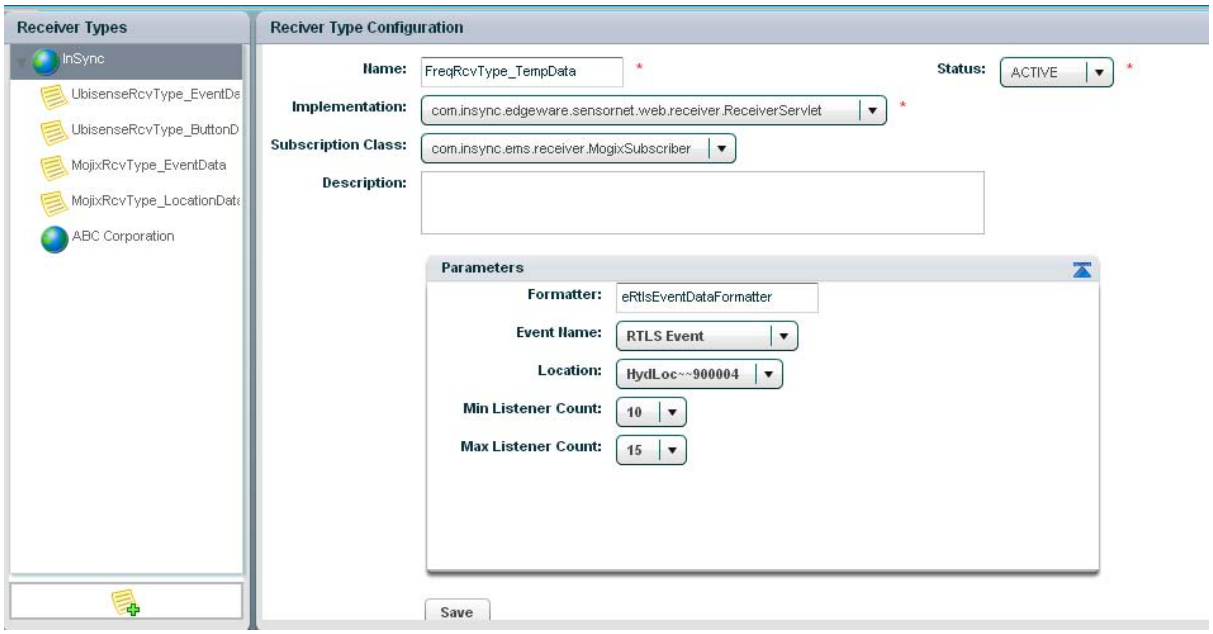
Each Receiver type is a java implementation which understands a set of parameters- event name to be raised and the formatter to be used.



iApp supports HTTP, Webservices, WiBox and TCP/IP protocol modes.

Create a Receiver Type

- Select the enterprise organization
- Click



Attribute	Description
Name	Unique name for the receiver type within each Enterprise.
Status	Active / InActive
Implementation	Java implementation of the receiver type. Select from the drop down.
Subscription Class	Is to subscribe to an external system from where the data will be posted to sensornet. This is to register the external system. **Note:** The EMS->Receiver page will have a Subscription tab from where the subscription parameters can be set, if the Receiver Type is configured with the Subscription class .
Description	Short description
Parameters	
Formatter	XSLT type formatters are supported by the receiver framework.
Event Name	Event name to be raised.
Location	Location to be overridden in the final event
Min Listener Count	Minimum listener threads to be created while processing the request
Max Listener Count	Maximum listener threads to be created while processing the request.

4.25.6 iApp Export Import Overview

Export and Import are two important features provided by IApp. These are useful to move objects from one instance of IApp installation to another instance.

- Export / Import are provided in IApp at the following functions.

- Configuration
- Data schemes
- Formatters
- Event Management Service
- Event Handlers
- Catalogs
- Image Map.

- Configuration Export / Import is explained in section 4.1.6
- Data Schemes export / import is explained in section 5.3.2
- Formatters export / import is explained in section 5.4.2
- Event Management Service export / import is explained in section 5.5.2
- Event handlers Export / Import are explained in IApp v3.3.1 Administration Manual.
- Catalogs export / import are explained in IApp v3.3.1 Administration Manual
- Image Map Export / Import is explained in section 4.1.6

Note: For all Imports happening it is required that the Enterprise Organization structure is in place before the import happens.

Import Pre-Requisites and Sequence of Import

1. **EMS Import**
   a. Enterprise Organization structure is in place
   b. Import Catalogs
   c. Import Event Handlers
   d. Import the configuration
2. **Configuration Export**
   e. Enterprise Organization structure is in place
   f. Import Catalogs
3. **Event Handlers Import**
   g. Enterprise Organization structure is in place
   h. Import Catalogs
4. **Catalog Import**
   i. Enterprise Organization structure is in place

Note: Creating an Enterprise organization creates a set of catalogs for that enterprise organization as explained in the Administration manual. If there are custom catalogs created for an enterprise and if used in EMS then it is recommended that the Catalogs are imported as well.

5 Device Types

5.1 Introduction

This document describes the Device Types available in RFID IApp.

The purpose is to document the different Config and Command Properties available for each Device Types.

5.2 Device Types

The Device Types available in IApp are

- Alien
- AWID MPR
- Cognex2D
- EMS
- Identec
- IdentecHandheld
- IdentecHandheldV2
- IdenteclportMB
- Impinj
- ImpinjRevolution
- Intelleflex
- Intermec
- Mojix
- MojixALE
- Portable
- Omron PLC
- RFID Printer (Zebra)
- Scale
- SIGN
- Sirit
- SmartId
- Standard Printer
- Symbol – Matrics
- ThingMagic
- Triax PLC
- Webcam
- TROVAN

Each device type has a set of Config and Command Properties.

Config Property – These are properties required for the setup and fine tuning of the device. The set of properties are based on the implementation of the device type. Each device type has its own set of properties defined.

Command Properties – These are commands which are understood by that particular device. The commands can be static commands or run time commands that are understood by the device implementation.

5.2.1 Alien

Command Properties

Name	Value	Comments
Read	get TagList	Read tags command to be sent to reader
Quit	Quit	Stop reading command to be sent to reader
Protocol	GEN2	Protocol parameter in the read command (Example: GEN2, EPC1, EPC0,ISO18000-6B)

Config Properties

Name	Value	Comments
Use Device Time	Y	Some devices send the read time, when they send use that time as observed time
Send Read Event	Y	When this is set to Y, the TagRead event will be sent to SensorNet. You can set this to Y when you don't want to send all the tag reads to SensorNet if you already have EdgeProcess generating application events
Antenna Count	4	How many antennas are connected to this reader
Antennas	As defined by the system 0,1,2,3,4	IApp allows antennas to be used as virtual readers. When you create a virtual reader you specify the antenna id's that you want this device to report for.
Command Prompt	Alien	Prompt for command line communication. Default value is "Alien". As showing in command window as "Alien>" for end user to type in command.
Connection Timeout	36000	Timeout in milli seconds before timing out whenever IApp tries to connect the reader
IP Address	10.0.1.24	IP Address of the reader
Login ID	alien	Login ID to be sent to reader before reader starts accepting further commands
Password	Password	Password for authentication along with Login ID
Observed Count	1	How many times should the reader see a tag to report it. Minimum is 1. Sometimes to make sure that we do not read stray reads we make the Observe Count as 2 or 3 or 4. This is to confirm the reads are accurate.
Port Number	23	Port Number where reader accepts the commands and sends the data.
Read Time Threshold	60000	This is the time in milli seconds to before a inactive device thread is restarted. Each device in IApp has a thread to read tags. If for some reason this thread is dead, then IApp starts a new thread after this time
Reader Response Wait time	250	Time in milli seconds to wait after a command is sent to reader for reading the response. If there is no response from the reader, then IApp gives up and completes the read cycle.
RF Attenuation	10	Attenuating the RF signal is a way of changing its power. By changing the attenuation you can increase or decrease the read range
RF Attenuation Max Limit	150	Maximum value allowed by the local country laws.

5.2.2 AWID MPR

Command Properties

Name	Value	Comments
Protocol	GEN2	Protocol parameter in the read command (Example: GEN2, EPC1, EPC0,ISO18000-6B)

Config Properties

Name	Value	Comments
Use Device Time	N	Some devices send the read time, when they send use that time as observed time. If set to N then take the clock time
Try Times	3	Number of times each command is tried by the reader if previous attempts are unsuccessful
Send Read Event	Y	When this is set to Y, the TagRead event will be sent to SensorNet. You can set this to Y when you don't want to send all the tag reads to SensorNet if you already have EdgeProcess generating application events
Reader Response Wait time	250	Time in milli seconds after a command is sent to reader but there is no response from the reader, then IApp gives up and completes
Read Time Threshold	60000	This is the time in milli seconds before a inactive device thread is restarted. Each device in IApp has a thread to read tags, for some reason this thread is dead, then IApp starts a new thread after this time
Port Number	4000	Port Number where reader accepts the commands and sends the data
Observed Count	1	How many times should the reader see a tag to report it. Minimum is 1. Sometimes to make sure that we do not read stray reads we make the Observe Count as 2 or 3 or 4. This is to confirm the reads are accurate.
IP Address	10.0.1.24	IP Address of the reader
Connection Timeout	36000	Timeout in milli seconds before timing out whenever IApp tries to connect the reader
Antennas	As defined by the system 0,1,2,3,4	IApp allows antennas to be used as virtual readers. When you create a virtual reader you specify the antenna id's that you want this device to report for.

5.2.3 Cognex2D

Command Properties

Name	Value	Comments
Read 2D Data	SM""0	Command to start reading the data
Poll 2D Data	SE8	Command to request for reading data. Used in SemiAuto mode

Change Mode To XML	PUT XML 1	After login, command sent to change to read mode

Config Properties

Name	Value	Comments
Read Time Threshold	60000	This is the time in milli seconds before a inactive device thread is restarted. Each device in IApp has a thread to read tags, for some reason this thread is dead, then IApp starts a new thread after this time
Login ID	Admin	Login ID to be sent to reader before reader starts accepting further commands
Password		Password for authentication along with Login ID
IP Address	10.0.1.24	IP Address of the reader
Port Number	23	Port Number where reader accepts the commands and sends the data
Connection timeout	60000	Timeout in milli seconds before timing out whenever IApp tries to connect the reader
Antennas	As defined by the system 0,1,2,3,4	IApp allows antennas to be used as virtual readers. When you create a virtual reader you specify the antenna id's that you want this device to report for.

5.2.4 EMS

Command Properties

Name	Value	Comments
Write To a Memory Block	start=[] data=[]	Command to write tag data to tag memory at a specified block number, this is supported in only GEN2
Read From a Memory Block		Command to read tag data from a specific block number, this is supported in GEN2 mode only
Protocol	GEN2	Protocol parameter in the read command (Example: GEN2, EPC1, EPC0,ISO18000-6B)

Config Properties

Name	Value	Comments
Use Device Time	N	Some devices send the read time, when they send use that time as observed time. If set to N then take the clock time
Send Read Event	Y	When this is set to Y, the TagRead event will be sent to SensorNet. You can set this to Y when you don't want to send all the tag reads to SensorNet if you already have EdgeProcess generating application events
Reader Response Wait	250	Time in milli seconds after a command is sent to reader but there is no response from the reader, then IApp gives up and completes
Read Time	60000	This is the time in milli seconds before a inactive

Threshold		device thread is restarted. Each device in IApp has a thread to read tags, for some reason this thread is dead, then IApp starts a new thread after this time
Observed Count	1	How many times should the reader see a tag to report it. Minimum is 1. Sometimes to make sure that we do not read stray reads we make the Observe Count as 2 or 3 or 4. This is to confirm the reads are accurate.
Serial Port	COM1	Serial port connect to the reader that accept reads or send commands
Parity	EVEN	Serial communication parameter.
Flow Control	NONE	Serial communication parameter.
Databits	7	Serial communication parameter.
Stopbits	2	Bits at which there is no data available
BaudRate	9600	Serial communication parameter.
Antennas	As defined by the system 0,1,2,3,4	IApp allows antennas to be used as virtual readers. When you create a virtual reader you specify the antenna id's that you want this device to report for.

5.2.5 Impinj

Command Properties

Name	Value	Comments
Protocol	GEN2	Protocol parameter in the read command (Example: GEN2, EPC1, EPC0,ISO18000-6B)

Config Properties

Name	Value	Comments
Use Device Time	Y	Some devices send the read time, when they send use that time as observed time. If set to N then take the clock time
Tx Power	188	value to set the modem RF-transmit power
Session	1	The modem uses when inventorying tags
Send Read Event	Y	When this is set to Y, the TagRead event will be sent to SensorNet. You can set this to Y when you don't want to send all the tag reads to SensorNet if you already have EdgeProcess generating application events
Regulatory Region	0	Modem checks the specified regulatory region against its hardware and firmware capabilities prior to accepting the command
Antennas	As defined by the system 0,1,2,3,4	IApp allows antennas to be used as virtual readers. When you create a virtual reader you specify the antenna id's that you want this device to report for.
Auto Set	2	serial port parameter
Connection Timeout	60000	Timeout in milli seconds before timing out whenever IApp tries to connect the reader
Estimate Tags	100	Hint to the expected tag population size. Valid range 0-65535
Estimate Time	500	In milli seconds range 0-65535. Hint to the expected

		tag time in field.
Inventory Reporting Mode	0	
Inventory Search Mode	1	
IP Address		IP Address of the reader
Mode ID	2	
Observed Count	1	How many times should the reader see a tag to report it. Minimum is 1. Sometimes to make sure that we do not read stray reads we make the Observe Count as 2 or 3 or 4. This is to confirm the reads are accurate.
Port Number	49380	Port Number where reader accepts the commands and sends the data
Read Time Threshold	60000	This is the time in milli seconds before a inactive device thread is restarted. Each device in IApp has a thread to read tags, for some reason this thread is dead, then IApp starts a new thread after this time
Reader Response Wait time	250	Time in milli seconds after a command is sent to reader but there is no response from the reader, then IApp gives up and completes

5.2.6 ImpinjRevolution

Command Properties

Name	Value	Comments

Config Properties

Name	Value	Comments
Use Device Time	Y	Some devices send the read time, when they send use that time as observed time. If set to N then take the clock time
Tx Power	61	value to set the modem RF-transmit power
RO Spec	1	
Send Read Event	Y	When this is set to Y, the TagRead event will be sent to SensorNet. You can set this to Y when you don't want to send all the tag reads to SensorNet if you already have EdgeProcess generating application events
Connection Timeout	2000	Timeout in milli seconds before timing out whenever IApp tries to connect the reader
IP Address	192.168.1.110	IP Address of the reader
Port Number	5048	Port Number where reader accepts the commands and sends the data
Reader Response Wait time	500	Time in milli seconds after a command is sent to reader but there is no response from the reader, then IApp gives up and completes

5.2.7 Intelleflex

Command Properties

Name	Value	Comments
Protocol	GEN2	Protocol parameter in the read command (Example: GEN2, EPC1, EPC0,ISO18000-6B)

Config Properties

Name	Value	Comments
Use Device Time	N	Some devices send the read time, when they send use that time as observed time. If set to N then take the clock time
Send Read Event	Y	When this is set to Y, the TagRead event will be sent to SensorNet. You can set this to Y when you don't want to send all the tag reads to SensorNet if you already have EdgeProcess generating application events
Return Link Type	1	
Return Link Rate	2	
Antenna Count	4	How many antennas are connected to this reader
Antennas	As defined by the system 0,1,2,3,4	IApp allows antennas to be used as virtual readers. When you create a virtual reader you specify the antenna id's that you want this device to report for.
Connection Timeout	60000	Timeout in milli seconds before timing out whenever IApp tries to connect the reader
Forward link rate	1	
Inventory Type	1	
IP Address	10.0.1.24	IP Address of the reader
Observed Count	1	How many times should the reader see a tag to report it. Minimum is 1. Sometimes to make sure that we do not read stray reads we make the Observe Count as 2 or 3 or 4. This is to confirm the reads are accurate.
Port Number	80	Port Number where reader accepts the commands and sends the data
Power	30	
Qval	4	
Read Protocol	C3	
Read Time Threshold	60000	This is the time in milli seconds before a inactive device thread is restarted. Each device in IApp has a thread to read tags, for some reason this thread is dead, then IApp starts a new thread after this time
Reader Response Wait Time	250	Time in milli seconds after a command is sent to reader but there is no response from the reader, then IApp gives up and completes the read cycle

5.2.8 Intermec

Command Properties

Name	Value	Comments
Write to a tag from antenna	WRITE EPCID=H[TagID] WHERE ANTENNA=[Antenna ID]	Command to write tag id on the tag from a particular antennna
Write To a tag	WRITE EPCID=H[Tag ID]	Command to write tag id on the tag from any antenna
Write to a memory bank from Antenna	WRITE HEX([Bank ID]:[Address], [Length])=H[Data] WHERE ANTENNA=[Antenna ID] AND EPCID=H[Tag ID]	Command to write tag data to a particular memory bank using a particular antenna. Used only in GEN2
Write to a memory bank	WRITE HEX([Bank ID]:[Address], [Length])=H[Data] WHERE EPCID=H[Tag ID]	Command to write tag data to a particular memory bank using any antenna. Used only in GEN2
Read from a memory bank from antenna	READ HEX([Bank ID]:[Address], [Length]) WHERE ANTENNA=[Antenna ID] AND EPCID=H[Tag ID]	Command to read tag data from a particular memory bank using a particular antenna
Read from a memory bank	READ HEX([Bank ID]:[Address], [Length]) WHERE EPCID=H[Tag ID]	Command to read tag data from a particular memory bank using any antenna
Read from a antenna	READ EPCID,ANTENNA WHERE ANTENNA=[Antenna ID]	Command to read a tag from a particular antenna
Read from all antennas	READ EPCID,ANTENNA	Command to read tags from all antennas
Protocol	GEN2	Protocol parameter in the read command (Example: GEN2, EPC1, EPC0,ISO18000-6B)
Kill a tag from antenna	WRITE EPCID=H[TagID] WHERE ANTENNA=[AntennaID]	Command to kill a tag so that it can not be read or written again, using a particular antenna
Kill a tag	WRITE EPCID=H[Tag ID]	Command to kill a tag so that it can not be read or written again, using any antenna

Config Properties

Name	Value	Comments
Use Device Time	N	Some devices send the read time, when they send use that time as observed time
Reader Response Wait Time	250	Time in milli seconds after a command is sent to reader but there is no response from the reader, then IApp gives up and completes
Send Read Event	Y	When this is set to Y, the TagRead event will be sent to SensorNet. You can set this to Y when you don't want to send all the tag reads to SensorNet if you already have EdgeProcess generating application events
Read Time Threshold	60000	This is the time in milli seconds before a inactive device thread is restarted. Each device in IApp has a thread to read tags, for some reason this thread is dead, then IApp starts a new thread after this time
Port Number	8080	Port Number where reader accepts the commands

		and sends the data
Observed Count	1	How many times should the reader see a tag to report it. Minimum is 1. Sometimes to make sure that we do not read stray reads we make the Observe Count as 2 or 3 or 4. This is to confirm the reads are accurate.
IP Address	10.0.1.24	IP Address of the reader
Connection timeout	60000	Timeout in milli seconds before timing out whenever IApp tries to connect the reader
Antennas	As defined by the system 0,1,2,3,4	IApp allows antennas to be used as virtual readers. When you create a virtual reader you specify the antenna id's that you want this device to report for.
Antenna Count	4	How many antennas are connected to this reader

5.2.9 Identec

Command Properties

Name	Value	Comments
Protocol	GEN2	Protocol parameter in the read command (Example: GEN2, EPC1, EPC0,ISO18000-6B)

Config Properties

Name	Value	Comments
Use Device Time	N	Some devices send the read time, when they send use that time as observed time
Send Read Event	Y	When this is set to Y, the TagRead event will be sent to SensorNet. You can set this to Y when you don't want to send all the tag reads to SensorNet if you already have EdgeProcess generating application events
IP Address	http://localhost:8080/processzone/services/ILRServices	IP Address of the reader

5.2.10 Identec Handheld

Command Properties

Name	Value	Comments
Protocol	GEN2	Protocol parameter in the read command (Example: GEN2, EPC1, EPC0,ISO18000-6B)

Config Properties

Name	Value	Comments
Use Device Time	Y	Some devices send the read time, when they send use that time as observed time
Send Read	Y	When this is set to Y, the TagRead

Event		event will be sent to SensorNet. You can set this to Y when you don't want to send all the tag reads to SensorNet if you already have EdgeProcess generating application events
IP Address	http://localhost:8080/processzone/services/ILRHandheldServices	IP Address of the reader

5.2.11 IdentecHandheldV2

Command Properties

Name	Value	Comments
Protocol	GEN2	Protocol parameter in the read command (Example: GEN2, EPC1, EPC0,ISO18000-6B)

Config Properties

Name	Value	Comments
Use Device Time	Y	Some devices send the read time, when they send use that time as observed time
Send Read Event	Y	When this is set to Y, the TagRead event will be sent to SensorNet. You can set this to Y when you don't want to send all the tag reads to SensorNet if you already have EdgeProcess generating application events
IP Address	http://localhost:8080/processzone/services/ILRHandheldServices	IP Address of the reader

5.2.12 Identec portMB

Command Properties

Name	Value	Comments
Protocol	GEN2	Protocol parameter in the read command (Example: GEN2, EPC1, EPC0,ISO18000-6B)

Config Properties

Name	Value	Comments
Use Device Time	Y	Some devices send the read time, when they send use that time as observed time
Send Read Event	Y	When this is set to Y, the TagRead event will be sent to SensorNet. You can set this to Y when you don't want to send all the tag reads to SensorNet if you already have EdgeProcess generating application events

IP Address	http://localhost:8080/processzone/services/ILRBeaconWebService	IP Address of the reader

5.2.13 Mojix

Command Properties

Name	Value	Comments

Config Properties

Name	Value	Comments
WSDL_URL	C:\insync\RFIDIApp\MojixTagsApi.wsdl	Wsdl file location
URL	http://localhost:8080/mojixserver/services/MojixStar1000	Address of web service
Data_Type	Tlh	“tlh” for inventory reads “ddd” for dock door reads (note must all be lower case)
CALLBACK_URI	MJ	Callback url for ale interface

5.2.14 MojixALE

Command Properties

Name	Value	Comments

Config Properties

Name	Value	Comments
URL	http://localhost:8080/processzone	Address of web service
Data_Type	tlh	“tlh” for inventory reads “ddd” for dock door reads (note must all lower case)
CALLBACK_URI	MJ	Callback url for ale interface
Reader Response Wait time	20	Time in milli seconds after a command is sent to reader but there is no response from the reader, then IApp gives up and completes the read cycle

5.2.15 Portable

Command Properties

Name	Value	Comments
Protocol	GEN2	Protocol parameter in the read command (Example: GEN2, EPC1, EPC0,ISO18000-6B)

Config Properties

Name	Value	Comments
Use Device Time	Y	Some devices send the read time, when they send use that time as observed time
Tag List Start	TagList	XML element tag for list of reads
Tag Group Start	DeviceReads	Root XML element tag
Tag Element Start	TagId	XML element tag for each read
Send Read Event	Y	When this is set to Y, the TagRead event will be sent to SensorNet. You can set this to Y when you don't want to send all the tag reads to SensorNet if you already have EdgeProcess generating application events
IP Address		IP Address of the reader. Each reader is a http client to the device driver, this field is used for identity reader not used to communicate with the reader

Sample message from reader:

```
 <DeviceReads>
<DeviceName>....</DeviceName>
<TimeStamp>....</TimeStamp>
<Location>....</Location >
<TagList>
                    <TagId>.....</TagId>
 <TagId>.....</TagId>
</TagList>
<Batch> …… </Batch>
 </DeviceReads>
```

5.2.16 Omron PLC

Command Properties

Name	Value	Comments

Config Properties

Name	Value	Comments
SerialPort	COM1	Name of serial port to send command and receive signals
Baudrate	9600	Serial port parameter
Databits	7	Serial port parameter
Stopbits	2	Serial port parameter
FlowControl	NONE	Serial port parameter
Parity	EVEN	Serial port parameter
ChangeDuration	1000	In milli seconds. Used to filter noise signals. Only report when signal stay changed longer than this time window.
HeartBeatInterval	60000	Time interval to send PLC read, only apply when support heartbeat is true
HeartBeatSupport	False	Send PLC read if it is true regardless if signal value changed or not
Observed Count	1	How many times should the reader see a tag to report it. Minimum is 1. Sometimes to make sure that we do not read stray reads we make the Observe Count as 2 or 3 or 4. This is to confirm the reads are accurate.
PollingInterval	500	Frequency to check plc signals
Read Time Threshold	60000	This is the time in milli seconds before a inactive device thread is restarted. Each device in IApp has a thread to read tags, for some reason this thread is dead, then IApp starts a new thread after this time
Reader Response Wait time	250	Time in milli seconds after a command is sent to reader but there is no response from the reader, then IApp gives up and completes the read cycle
SignalCount	8	PLC register size, number of pin

5.2.17 RFID Printer

Command Properties

Name	Value	Comments

Print-Serials	^XA^RB96,8,3,3,20,24,38^RMN^RFw,e^FD48,1,6,114415,0,[SerialNumber]^FS^XZ^XA^SZ2^JMA^MCY^PMN^PW806~JSN^JZY^LH0,0^LRN^XZ~DGR:SSGFX000.GRF,920,10,:Z64:eJxt07Fq3EAQBuBZaZCWMBy6wrDFYSt9ig3h4EoZH6n8Cga5cOcindN5YxtXepI8hZI7TF4g/T1CHsCw3hv9ey4uI4Q+fq2k3VlElCoGmqr++wqtaAZ9oxK302VUlHSV4n0VFHcrFdf03amsJRGV81RXKt+TeZg0ktmo2vCBfumrr0Mcr1XnJuxa1UVJ3uvnHgpyjWYvnNRpJmQbvfssJMisiOumTBiyM2QvcprH8QzP/q7OHFSceY8ZnGSFJbWY3zKcq9rxU9hgRTc/HrHK28dCtVis0xT3JXL5x6oq2nYOXRt6rzLjtu+nnu6Gj+ju56cv6LgZfBZnldJkcVYhDqrYHTIL1THvZTUMEDNDIlUWZ1kpIPffrIQaOYgN5MVgHZ6zWiHo7v5unHQfI9QxZQl1k0bOCkI9xAcJtQqTMnQyZRAxNZBQ7jhDRcosdJxVKRNIINYDLUo5rdfrLp1fO4oxhnTGfzSfN2aeKu/IUdX7cfp7veu4ip/70hnrqIQ39INrYQ==:51B3^XA^DFR:SSFMT000.ZPL^FS^FO19,200^BY2,2.5^B3N,N,102,N,N^FN1^FS^FT212,324^CI0^A0N,23,32^FN2^FS^FT58,465^A0N,102,140^FDRFID^FS^FO38,358^GB395,141,14^FS^FT464,381^A0N,34,47^FDSerial^FS^FT464,465^A0N,102,133^FN3^FS^FT45,538^A0N,39,54^FDOVERPACK BOX^FS^FT70,189^ABN,33,21^FN4^FS^FO327,379^XGR:SSGFX000.GRF,1,1^FS^XZ^XA^XFR:SSFMT000.ZPL^FN1^FD30386FBBC0000000000[0xSerialNumber]^FS^FN2^FD30386FBBC0000000000[0xSerialNumber]^FS^FN3^FD[SerialNumber]^FS^FN4^FD30386FBBC0000000000[0xSerialNumber]^PQ1,0,1,Y^XZ^XA^IDR:SSFMT000.ZPL^XZ^XA^IDR:SSGFX000.GRF^XZ	Command to print tags in serial numbers. The command has variables SerialNumber embedded in it, at run time this variable will be replaced with a system generated serial number
Print-Odds	^XA^RB96,8,3,3,20,24,38^RMN^RFw,e^FD48,0,6,114415,0,[SerialNumber]^FS^XZ^XA^SZ2^JMA^MCY^PMN^PW806~JSN^JZY^LH0,0^LRN^XZ~DGR:SSGFX000.GRF,920,10,:Z64:eJxt0rGO00AQBuDx2BpbYCU2iOACpL0U1FtdG+eIrr4HoHBAgjZvwCZI0QlR8AQ8AE+xR0HPA6DLY1AglvXMrKUIxrH06c/anh0bINbqBFLlj9+qGTxU7SDzooF/sTL/BCwrh3CqWFT6V4VoBoSs+hlkH1hVXL5nNfEWazfK9jMwIv/eW36Icd42/ajt3nWdKIeq4gevcyhELzHKsAio5uyKEDUrSqr13zJlVBay7g3O9dp1Pv/UcXax/3ib9OWzdGDcA2d76XSh/TXDIz9wz51d9Heyt265lV0W9eW7koW4fEE6obemlqm516aTSfpruxPZb1aH33xvnY78oMsAjpMOlSI71kmHSVSocpwU/qhiLyoiShnilOWqAidRpqox065qSqoQVB2BV01ZM2UNQi96fv9Ts1UIul9L+jWBQTCigfS7ggGTeoJGhUmeQAfjMOksk2FljiDNHpOAQEaUA0J6R6RCPngs8Ujj4Kxt25v2afu4gRCCj2f4BZvNJovnRrf5b+XjuuDP9J/6OpYbu+NVUX8BClFl9w==:D74C^XA^DFR:SSFMT000.ZPL^FS^FO32,200^BY2,2.5^B3N,N,102,N,N^FN1^FS^FT222,324^CI0^A0N,23,32^FN2^FS^FT45,452^A0N,102,140^FDRFID^FS^FO25,358^GB395,141,14^FS^FT464,381^A0N,34,47^FDSerial^FS^FT464,465^A0N,102,133^FN3^FS^FT32,544^A0N,45,62^FDINTERMEDIATE	Same as above except the tag id are in odds like 1,3,5,7

	BOX^FS^FT83,189^ABN,33,21^FN4^FS^FO313 ,379^XGR:SSGFX000.GRF,1,1^FS^XZ^XA^XFR :SSFMT000.ZPL^FN1^FD30186FBBC00000000 00[0xSerialNumber]^FS^FN2^FD30186FBBC00 00000000[0xSerialNumber]^FS^FN3^FD[SerialN umber]^FS^FN4^FD30186FBBC0000000000[0x SerialNumber]^PQ1,0,1,Y^XZ^XA^IDR:SSFMT0 00.ZPL^XZ^XA^IDR:SSGFX000.GRF^XZ	
Print – Evens	^XA^RB96,8,3,3,20,24,38^RMN^RFw,e^FD48,0, 6,114415,0,[SerialNumber]^FS^XZ^XA^SZ2^JM A^MCY^PMN^PW806~JSN^JZY^LH0,0^LRN^X Z~DGR:SSGFX000.GRF,920,10,:Z64:eJxt0rGO 00AQBuDx2BpbYCU2iOACpL0U1FtdG+eIrr4Ho HBAgjZvwCZI0QlR8AQ8AE+xR0HPA6DLY1Agl vXMrKUIxrH06c/anh0bINbqBFLlj9+qGTxU7SDz ooF/sTL/BCwrh3CqWFT6V4VoBoSs+hlkH1hVX L5nNfEWazfK9jMwIv/eW36Icd42/ajt3nWdKIeq4 gevcyhELzHKsAio5uyKEDUrSqr13zJlVBay7g3O 9dp1Pv/UcXax/3ib9OWzdGDcA2d76XSh/TXDIz 9wz51d9Heyt265lV0W9eW7koW4fEE6obemlqm 516aTSfpruxPZb1aH33xvnY78oMsAjpMOlSI71k mHSVSocpwU/qhiLyoiShnilOWqAidRpqox065qS qoQVB2BV01ZM2UNQi96fv9Ts1UIul9L+jWBQT CigfS7ggGTeoJGhUmeQAfjMOksk2FljiDNHpOA QEaUA0J6R6RCPngs8Ujj4Kxt25v2afu4gRCCj2f 4BZvNJovnRrf5b+XjuuDP9J/6OpYbu+NVUX8B ClFl9w==:D74C^XA^DFR:SSFMT000.ZPL^FS^F O32,200^BY2,2.5^B3N,N,102,N,N^FN1^FS^FT2 22,324^CI0^A0N,23,32^FN2^FS^FT45,452^A0N ,102,140^FDRFID^FS^FO25,358^GB395,141,14 ^FS^FT464,381^A0N,34,47^FDSerial^FS^FT464 ,465^A0N,102,133^FN3^FS^FT32,544^A0N,45,6 2^FDINTERMEDIATE BOX^FS^FT83,189^ABN,33,21^FN4^FS^FO313 ,379^XGR:SSGFX000.GRF,1,1^FS^XZ^XA^XFR :SSFMT000.ZPL^FN1^FD30186FBBC00000000 00[0xSerialNumber]^FS^FN2^FD30186FBBC00 00000000[0xSerialNumber]^FS^FN3^FD[SerialN umber]^FS^FN4^FD30186FBBC0000000000[0x SerialNumber]^PQ1,0,1,Y^XZ^XA^IDR:SSFMT0 00.ZPL^XZ^XA^IDR:SSGFX000.GRF^XZ	Same as above except the tag id are in evens like 2,4,6,8

Config Properties

Name	Value	Comments
Port Number	9100	Port Number where the printer accepts the commands and prints the data
IP Address		IP Address of the printer
Connection Timeout		Timeout in milli seconds before timing out whenever IApp tries to connect the printer

5.2.18 ThingMagic

Command Properties

Name	Value	Comments
Write to a tag	UPDATE tag_id SET id=[Tag Id] WHERE protocol_id='[Protocol]' AND antenna_id=[Antenna Id]	Command to write tag id
Write to a memory block for Gen2	UPDATE tag_data SET data=[Data] WHERE protocol_id='[Protocol]' AND antenna_id=[Antenna Id] AND block_number=[Block Number] AND block_count=[Block Count(1-127)] AND mem_bank=[Memory Bank(0-3)]	Command to write tag data to a particular memory block in GEN2 mode
Write to a memory block	UPDATE tag_data SET data=[Data] WHERE id=[Tag Id] AND blocknum=[Block Number] AND protocol_id='[Protocol]'	Command to write tag data to a particular memory block
Write to a memory bank for GEN2	UPDATE tag_data SET data=[Data] WHERE protocol_id='[Protocol]' AND antenna_id=[Antenna Id] AND block_number=[Block Number] AND mem_bank=[Memory Bank(0-3)]	Command to write tag data to a particular memory bank in memory block in GEN2 mode
Unlock a Gen2 Tag	UPDATE tag_id SET locked=0, id=[Tag Id],password=[Access Password] WHERE protocol_id='[Protocol]' AND antenna_id=[Antenna Id] AND type=48	Command to unlock previously locked tag
Set Kill password	UPDATE tag_id SET password=[Kill Password] WHERE protocol_id='[Protocol]' AND antenna_id=[Antenna Id]	Command to set password to kill
Auto Cursor off	SET AUTO=off	Command to stop the auto-cursor
Auto Cursor On	SET AUTO c1000=on	Command to start auto-cursor
Close Cursor1	CLOSE c1000	Command to close a open cursor
Declare cursor 1	DECLARE c1000 CURSOR for SELECT id, antenna_id FROM tag_id WHERE protocol_id='[Protocol]' AND (antenna_id=1 OR antenna_id=2 OR antenna_id=3 OR antenna_id=4) SET time_out=500	Command to declare a cusror
Kill a tag	UPDATE tag_id SET killed=1, id=[Tag Id], password=[Kill Password], WHERE protocol_id='[Protocol]' AND antenna_id=[Antenna Id]	Command to kill a tag so that it cant be read or written again
Lock a memory block	UPDATE tag_data SET locked=1 WHERE id=[] AND blocknum=[Block Number(1-127)] AND protocol_id='[Protocol]'	Command to lock a particular memory block
Lock a tag	UPDATE tag_id SET locked=32, id=[Tag Id],password=[Access Password] WHERE	Command to lock a tag so that it cant be read without a password

	protocol_id='[Protocol]' AND antenna_id=[Antenna Id] AND type=48	
Read from a antenna	SELECT id FROM tag_id WHERE antenna_id=[Antenna Id] AND protocol_id='[Protocol]'	Command to read tag id from a particular antenna
Read from a memory bank for GEN2	SELECT data FROM tag_data WHERE block_number=[Block Number] AND mem_bank=[Memory Bank(0-3)] AND protocol_id='[Protocol]' AND antenna_id=[Antenna Id]	Command to read tag data from a particular memory bank in GEN2 mode
Read from a memory block	SELECT data FROM tag_data WHERE id=[Tag Id] AND blocknum=[Block Number] AND protocol_id='[Protocol]'	Command to read tag data from a particular memory block
Read from a memory block for GEN2	SELECT data FROM tag_data WHERE protocol_id='[Protocol]' AND antenna_id=[Antenna Id] AND block_number=[Block Number] AND block_count=[Block Count(1-127)] AND mem_bank=[Memory Bank(0-3)]	Command to read tag data from a particular memory block in GEN2
Read from all antennas	SELECT id FROM tag_id WHERE protocol_id='[Protocol]'	Command to read tags from all antennas
Set Access password for GEN2 tags	UPDATE tag_id SET password=[Access Password] WHERE protocol_id='[Protocol]' AND antenna_id=[Antenna Id] AND type=1	Command to set access password. When set, all read commands should be sent along with this password

Config Properties

Name	Value	Comments
Use Device Time	N	Some devices send the read time, when they send use that time as observed time
Send Read Event	Y	When this is set to Y, the TagRead event will be sent to SensorNet. You can set this to Y when you don't want to send all the tag reads to SensorNet if you already have EdgeProcess generating application events
Reader Response Wait Time	250	Time in milli seconds after a command is sent to reader but there is no response from the reader, then IApp gives up and completes the read cycle
Read Time Threshold	60000	This is the time in milli seconds before a inactive device thread is restarted. Each device in IApp has a thread to read tags, for some reason this thread is dead, then IApp starts a new thread after this time
Port Number	8080	Port Number where reader accepts the commands and sends the data
Observed	1	How many times should the reader

Count		see a tag to report it. Minimum is 1. Sometimes to make sure that we do not read stray reads we make the Observe Count as 2 or 3 or 4. This is to confirm the reads are accurate.
IP Address		IP Address of the reader
Connection time out	60000	Timeout in milli seconds before timing out whenever IApp tries to connect the reader
Antennas	As defined by the system 0,1,2,3,4	IApp allows antennas to be used as virtual readers. When you create a virtual reader you specify the antenna id's that you want this device to report for.
Antenna Count	4	How many antennas are connected to this reader

5.2.19 Triax PLC

Command Properties

Name	Value	Comments
CLEAR	C	Required to enable "AUTO" after any hard error even if the error was only momentary. It resets the fault memory if the fault no longer exists.
DEPOSIT	D [Horizontal] [Vertical] [Side]	Directs the retriever to move to the specified bin and to deposit a load in the bin
DUEL CYCLE	S[Station1] [Horizontal1] [Vertical1] [Side1] [Aisle1]	To pick up a load from a P&D and deposit it to a selected location in the structure followed by a retrieve from a selected location in the structure and depositing the load to a P&D
MOVE HOR	H [Horizontal]	Directs the retriever to move to horizontal address [Horizontal]
MOVE SHUTTLE	Z [Side]	Directs the retriever to center the shuttle
MOVE VERT	V [Vertical]	Directs the retriever to move to vertical address [Vertical]
PICKUP	P [Horizontal] [Vertical] [Side]	Directs the retriever to move to the specified bin address and pick up a load
RAW_COMMAND	[S]	Used to send any command
RESYNCHRONIZATION	C	Causes the retriever to resynchronize its primary addressing logic with its verification addressing logic
RETRIEVE	R[Station] [Horizontal] [Vertical] [Side] [Aisle]	To pick up a load from a selected location in the structure and deposit it to a deposit station
SELECT AISLE	C[Aisle]	

STORE	S[Station] [Horizontal] [Vertical] [Side] [Aisle]	To pick up a load from a pickup station and deposit the load to a selected location in the structure
TRANSMIT STATUS	X	Directs the retriever to transmit status information

Config Properties

Name	Value	Comments
SerialPort	COM1	Serial port name
BaudRate	9600	Serial port parameter
Databits	8	Serial port parameter
Stopbits	1	Serial port parameter
Parity	EVEN	Serial port parameter
FlowControl	XNOFFON	Serial port parameter
Retriever	1	Default aisle position

5.2.20 SIGN

Command Properties

Name	Value	Comments

Config Properties

Name	Value	Comments
Rows	4	Number of rows to display. Reduce this number when increase font size
IP Address		IP address for the device
Port Number	4000	Port number where device accepts commands
Connection Timeout	36000	Timeout in milli seconds before timing out whenever IApp tries to connect the device
Line 1		Default value for line 1
Line 2		Default value for line 2
Line 3		Default value for line 3
Line 4		Default value for line 4
Color 1	Green	Default color for line 1
Color 2	Green	Default color for line 2
Color 3	Green	Default color for line 3
Color 4	Green	Default color for line 4
Font size 1	5	Default font size for line 1
Font Size 2		Default font size for line 2
Font Size 3		Default font size for line 3
Font Size 4		Default font size for line 4

5.2.21 Scale

Command Properties

Name	Value	Comments
Read	W	Command to read the weight

Config Properties

Name	Value	Comments
SerialPort	COM1	Serial port name
BaudRate	9600	Serial port parameter
Databits	7	Serial port parameter
Stopbits	2	Serial port parameter
Parity	EVEN	Serial port parameter
FlowControl	NONE	Serial port parameter
Reader Response Wait Time	250	Time in milli seconds after a command is sent to reader but there is no response from the reader, then IApp gives up and completes the read cycle
Read Time Threshold	60000	This is the time in milli seconds before a inactive device thread is restarted. Each device in IApp has a thread to read tags, for some reason this thread is dead, then IApp starts a new thread after this time
Observed Count	1	How many times should the reader see a tag to report it. Minimum is 1. Sometimes to make sure that we do not read stray reads we make the Observe Count as 2 or 3 or 4. This is to confirm the reads are accurate.

5.2.22 SmartId

Command Properties

Name	Value	Comments

Config Properties

Name	Value	Comments
Use Device Time	N	Some devices send the read time, when they send use that time as observed time
Send Read Event	Y	When this is set to Y, the TagRead event will be sent to SensorNet monitoring screen. You can set this to Y when you don't want to send all the

		tag reads to SensorNet if you already have EdgeProcess generating application events
Serialport	COM1	Serial port name
BaudRate	115200	Serial port parameter
Databits	8	Serial port parameter
Stopbits	2	Serial port parameter
Parity	NONE	Serial port parameter
FlowControl	NONE	Serial port parameter
Reader Ports	1	
Power Control	30	
Antennas	As defined by the system 0,1,2,3,4	IApp allows antennas to be used as virtual readers. When you create a virtual reader you specify the antenna id's that you want this device to report for.

5.2.23 Sirit

Command Properties

Name	Value	Comments
Write User Data	tag.write_user_data(user_data=[Data],lock_type=[Lock Type], tag_id=[Tag Id] , pwd=[Password]), antenna=[Antenna Id])	Command write user data in the memory
Write to a tag	tag.write_id(new_tag_id=[Tag Id	Command to write tag data
Unlock a GEN2 tag	tag.unlock(tag_id=[Tag Id] , pwd=[Password], unlock_fields=[Unlock Filelds])	Command to unlock previously locked tag
Set Kill Password	tag.write_kill_pwd(kill_pwd=[Kill Password], lock_type=[Lock Type], tag_id=[Tag Id] , pwd=[Password]), antenna=[Antenna Id])	Command to set password to kill a tag
Set Access password for GEN2 tags	tag.write_access_pwd(access_pwd=[Access Password], lock_type=[Lock Type], tag_id=[Tag Id] , pwd=[Password]), antenna=[Antenna Id])	Command to set password to read from a tag
Overwrite to a tag	tag.write_id(new_tag_id=[Tag Id], tag_id = [Old Tag Id])	Command to overwrite a tag id
Lock User Data	tag.lock_user_data(lock_type=[Lock Type])	Command to lock the user data in the memory
Lock a tag	tag.lock_id(lock_type=[Lock Type], lock_fields=[Lock Fields], tag_id=[Tag Id], pwd=[Password])	Command to lock a tag from reading
Kill a tag	tag.kill(kill_pwd=[Kill Password], tag_id=[Tag Id], antenna=[Antenna Id])	Command to kill a tag so that the tag cant be read or written again

Config Properties

Name	Value	Comments
Use Device Time	N	Some devices send the read time, when they send use that time as observed time
Send Read Event	Y	When this is set to Y, the TagRead event will be sent to SensorNet monitoring screen. You can set this to Y when you don't want to send all the tag reads to SensorNet if you already have EdgeProcess generating application events
Reader response Wait time	250	Time in milli seconds after a command is sent to reader but there is no response from the reader, then IApp gives up and completes
Protocol	ISOC ISOB SUPERTAG	Protocol parameter in the read command (Example: GEN2, EPC1, EPC0,ISO18000-6B)
Port Number	500007	Port Number where reader accepts the commands and sends the data
IP Address	10.0.1.24	IP Address of the reader
Antennas	As defined by the system 0,1,2,3,4	IApp allows antennas to be used as virtual readers. When you create a virtual reader you specify the antenna id's that you want this device to report for.

5.2.24 Standard Printer

Command Properties

Name	Value	Comments

Config Properties

Name	Value	Comments

5.2.25 Symbol – Matrics

Command Properties

Name	Value	Comments
read	Oper=queryEvents	Command to read the tags
raw	1	

Config Properties

Name	Value	Comments
Wait Time Between Query	2000	Time between open/close http connection
URL	http://localhost/cgi-bin/dataProxy?	url for http connection
Reader Response Wait Time	250	Time in milli seconds after a command is sent to reader but there is no response from the reader, then IApp gives up and completes the read cycle
Read Time Threshold	60000	This is the time in milli seconds before a inactive device thread is restarted. Each device in IApp has a thread to read tags, for some reason this thread is dead, then IApp starts a new thread after this time
Observed Count	1	How many times should the reader see a tag to report it. Minimum is 1. Sometimes to make sure that we do not read stray reads we make the Observe Count as 2 or 3 or 4. This is to confirm the reads are accurate.
Antenna Count	1	How many antennas are connected to this reader

5.2.26 TROVAN

Command Properties

Name	Value	Comments

Config Properties

Name	Value	Comments
Use Device Time	N	Some devices send the read time, when they send use that time as observed time
Serialport	COM1	Serial port name
BaudRate	19200	Serial port parameter
Databits	8	Serial port parameter
Stopbits	1	Serial port parameter
Parity	NONE	Serial port parameter
FlowControl	NONE	Serial port parameter
Send Read Event	Y	When this is set to Y, the TagRead event will be sent to SensorNet monitoring screen. You can set this to Y when you don't want to send all the tag reads to SensorNet if you already have EdgeProcess generating

		application events
Reader iD	01	Used in command
Buzzer Status	OF	Turn on/off buzzer
Antennas	As defined by the system 0,1,2,3,4	IApp allows antennas to be used as virtual readers. When you create a virtual reader you specify the antenna id's that you want this device to report for.

5.2.27 Webcam

Command Properties

Name	Value	Comments

Config Properties

Name	Value	Comments
SerialPort	COM1	Serial port name
BaudRate	9600	Serial port parameter
Databits	7	Serial port parameter
Stopbits	2	Serial port parameter
FlowControl	NONE	Serial port parameter
Parity	EVEN	Serial port parameter
CameraLocator	Vfw://0	*URL to construct this media*
Camera Ready Wait time	2000	*Command to wait for camera to get started*
Frames To Buffer	4	*How many frames will be stored in one buffer before saving to disk*
Image Resolution	1.0f	*Image quality*
Observed Count	1	How many times should the reader see a tag to report it. Minimum is 1. Sometimes to make sure that we do not read stray reads we make the Observe Count as 2 or 3 or 4. This is to confirm the reads are accurate.
Read Time Threshold	60000	This is the time in milli seconds before a inactive device thread is restarted. Each device in IApp has a thread to read tags, for some reason this thread is dead, then IApp starts a new thread after this time
Reader Response wait time	250	Time in milli seconds after a command is sent to reader but there is no response from the reader, then IApp gives up and completes

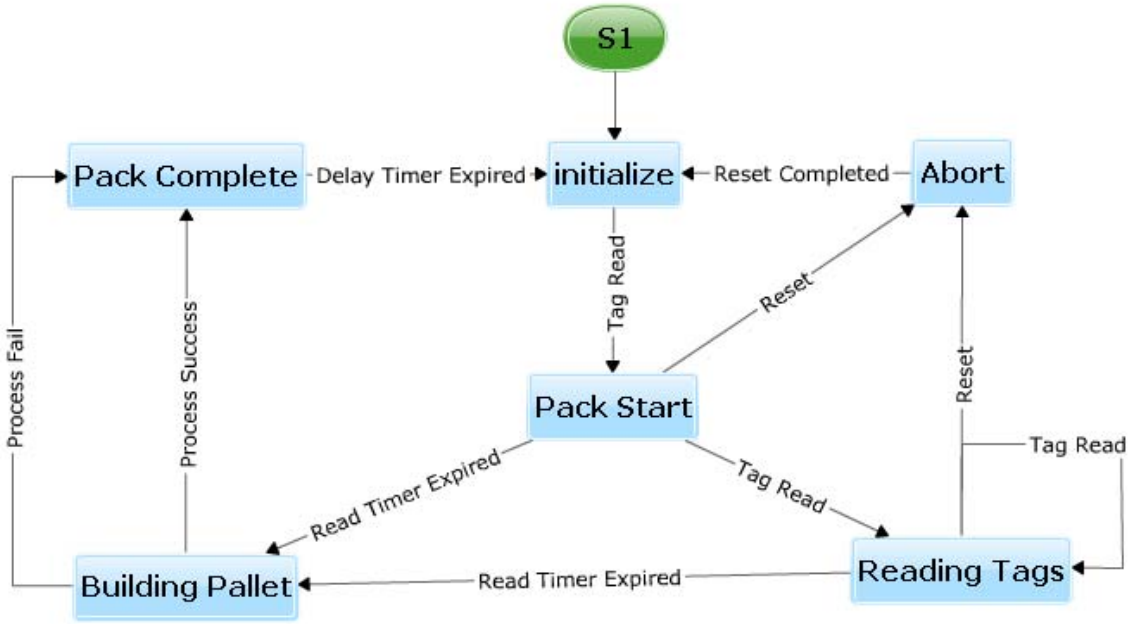
6 Edge Process

6.1 Introduction

An Edge Process is a means by which a physical process can be described in terms of that states, transitions and actions.

- A state stores information about the past, i.e. it reflects the input changes from the start to the current date.
- A transition is a state change and is described by a condition that needs to be fulfilled.
- An action is an activity that is to be performed.

The Edge Process hub forms the basis for all sensor-driven processes that allow collaboration across various locations. Once setup, the Edge Process hub acts as the filter to transform the raw data into events that are relevant to any business process. Acting as a tool for all parties in a business system, the Edge Process hub allows all groups to monitor, take inventory, and optimize business logic.



6.2 Edge Process Fundamentals

There are three fundamental concepts that are essential in the creation of an Edge Process: States, Events and Actions. IApp allows the user to define any arbitrary state(s) that maybe useful in describing a physical process. Once a state has been defined, IApp™Actions can be performed in any state and are handled by IApp™ Controlled Objects. An Action may result in an IApp event that are defined by the IApp™ Event Provider(s).

IApp has several pre-defined Controlled Objects each of which contains a list of actions that are available to be performed by a configured Edge Process.

IApp™ has a set of pre-defined Event names grouped together logically that can be used to effect an event transition from one state to another.

So in summary the following are the steps to configure an Edge Process using the IApp™ configuration:

1. Describe the process in terms of States, Actions and Events
2. Choose one of the provided Action Groups to perform the desired Action(s) in each State.
3. Based on the outcome of the Action(s), generate an event which causes the process to transition from the current state to another state.
4. Post appropriate events at key points in the execution of the process.

Notes: the following terms are used across the document

- Info log file: edgeware.log
- Debug log file: edgeware_debug.log

6.3 Device Action Groups

Action Group (Controlled Object)

IApp™ collects a set of related Actions into a group called Controlled Object. There are two kinds of Actions:

- Actions that execute commands on devices and perform work like
  - Turn RFID Reader ON/OFF
  - Change the Read mode of a RFID Reader
  - Flash a LED on a Light Stack

- Actions that aid in the performance of a physical process like
  - Validate if a Tag ID matches a predefined tag scheme
  - Associate a Parent Tag ID to a Child Tag ID
  - Post an Application Event to SensorNet with data collected in the process

Associated with each Action group (Controlled Object) is a set of configuration parameters that need to be specified.

6.3.1 AssociationProcessCO

This control object processes the scanned tags list which contains one parent tag id and others are its corresponding child tag ids. It will be used to form an association between parent and it's child items.

Actions: The following actions can be selected from the UI. Scripts cannot use these actions. All these actions take StateMachineContext as an input parameter.

Action	Description
Add Tags to UI	This action puts the scanned parent tag id and child tag ids into the list in the form of ReadHistory object. Before putting it checks the whether already ReadHistory with parent tag id or child tag ids exists. If so it won't add to the list otherwise it adds to the list.
Associate Parent to Children	This action checks in scanned list of tags there is only one parent tag id and number of scanned child tag ids equal to the expected count configured, if it matches then it will call the action "Post Process Success Event" otherwise it will call the action "Post Process Fail Event"
Clear Scanned Tags List	This action clears all tags scanned from the scanned tag list
Collect Unique Reads	This action loop through the all the scanned

	tag list, checks whether tag read time is set or not, if not then sets the tag read time and put the tag id as key and DataRead object as value in the hash map. After putting into the hash map it removes that entry from the scanned tag list
Disable The Device	This action **disable** all the devices associated to the corresponding edge process
Enable The Device	This action **enable** all the devices associated to the corresponding edge process
Extend the timer for unique tag	This action loops through the scanned tag list and check for each tag detail is there in the unique tag list or not. If it is not there then it “Reset Delay Wait Timer” action is called. And if tag read time is not set then it sets tag read time
Initialize	This action calls AssociationProcessCO parent EdgeProcessCO “Initialize” action and its local parent tag list is cleared.
Play Audio	This action plays the audio set in the parent EdgeProcessCO class
Post Application Event	This action post the event with name configured in edge process properties with scanned parent tag and child tag ids
Post Application Event For Single Tag	This action post the event with name configured in edge process properties for single tag by forming the string using the event parameters and with event type as “PROCESS”
Post Process Fail Event	This action set status of corresponding **EdgeProcessRuntime** as fail and calls related edge process **processStatus** method with parameter as integer constant which represent as fail
Post Process Status Event	**This action** calls related edge process **processStatus** method with status of **EdgeProcessRuntime** as it’s parameter
Post Process Success Event	This action set status of corresponding **EdgeProcessRuntime** as success and calls related edge process **processStatus** method with parameter as success constant
Post Tag Read Event	This action post **Tag Read** event, with event type as **DEVICEREAD** for the first entry in the scanned tag list and clears the scanned tag list.
Process PLC Read	Method corresponding to this action is empty. (No code inside the method)
Process Scale Read	Method corresponding to this action is empty. (No code inside the method)

Process Tag Read	Method corresponding to this action is empty. (No code inside the method)
Remove All Event Parameters	This action removes all the event parameters
Remove All Global Parameters	This action removes all the parameters which are used across the **EdgeProcessRuntime** objects
Remove All Parameters	Removes all entries from the _parameters in the parent EdgeProcessCO. (Note: Not able to get what type of information stored in this map)
Reset	This action reset all the variables of the control object, stops the reader timer, delay wait timer by calling "**Stop Read Timer**", "**Stop Delay Wait Timer**" respectively and clears the read history. Finally **resetCompleted** event is raised.
Reset Delay Wait Timer	This action resets the **Delay Timer** by calling the **Stop Delay Wait Timer** and **Start Delay Wait Timer**
Reset Read Timer	This action resets the **Read Timer** by calling the **Stop Read Timer** and **Start Read Timer**
Retrieve Context List from Tracking	This action gets the item details for each tag id from the Tracking Manager. (As EOL for Tracking Manager this action should be removed)
Start Delay Wait Timer	This action starts time for the **Delay Timer**
Start Read timer	This action starts time for the **Read Timer**
Start Reading	This action sets boolean **_startRead** to **true**
Start reset Timer	This action starts time for the **Reset Timer** and status of the **EdgeProcessRunTime** set as none
Stop Delay Wait Timer	This action stops the **Delay Timer**
Stop Read Timer	This action stops the **Read Timer**
Stop Reading	This action sets boolean **_startRead** to **false**
Stop Reset Timer	This action stops the **Reset Timer**
Transition to Init State	This action raises **toInitState** event through corresponding edge process
Transition to Next State	This action raises **toNextState** event through corresponding edge process
Validate PLC Read	Method corresponding to this action is empty. (No code inside the method)
Validate Scale Read	Method corresponding to this action is empty. (No code inside the method)
Validate single Tag read	This action gets the first entry from the scanned tag list which is not processed by checking for the **processed time** (if process time is zero then it is not processed) and

	validates the tag by calling the **validateTag** method passing the tag id. If it returns true then "**Post Process Success Event**" action will be called otherwise "**Post Process Fail Event**" action will be called
Validate Tag Reads	This action loops through the scanned tag list and for each tag is validated against the tag validations configured if scanned tag id is failed for one tag validation then it will be removed from the scanned tag list and added to the **_issueTagList** map (key as tag and DataRead as value)

Script Methods

Script methods are those which can be used in the PreScript and PostScript writing while configuring the EdgeProcess.

- init (): This method calls the super class EdgeProcessCO.init () method and sets the child encoders and parent coders for this control object for tag validations.
- getParentTag (): This method returns the either empty string or gets the first DataRead object in the parent tag list
- resetVariables (): This method calls the EdgeProcess.resetVariable () method and clears the parent tag list.
- setExpectedCount (): This method sets the expected count of Tag Reads
- getExpectedCount (): This method returns the expected count

Note: Apart from these we can use the super class EdgeProcessCO script methods also

6.3.2 ContextEdgeProcessCO

This control object doesn't have any actions on its own all the actions are derived from its super class **EdgeProcessCO**

Actions: The following actions can be selected from the UI. Scripts cannot use these actions. All these actions take StateMachineContext as an input parameter.

Action	Description
Add Tags to UI	This action puts the scanned parent tag id and child tag ids into the list in the form of ReadHistory object. Before putting it checks the whether already ReadHistory with parent tag id or scanned tag ids exists. If so it won't add to the list otherwise it adds to the list. (But note that in this action there is no setting of parent tag in the ReadHistory object is done)
Clear Scanned Tags List	This action clears all tags scanned from the scanned tag list
Collect Unique Reads	This action loop through the all the scanned tag list, checks whether tag read time is set or not, if not then sets the tag read time and put the tag id as key and DataRead object as

	value in the hash map. After putting into the hash map it removes that entry from the scanned tag list
Disable the device	This action **disable** all the devices associated to the corresponding edge process
Enable the device	This action **enable** all the devices associated to the corresponding edge process
Extend the timer for unique tag	This action loops through the scanned tag list and check for each tag detail is there in the unique tag list or not. If it is not there then it “Reset Delay Wait Timer” action is called. And if tag read time is not set then it sets tag read time
Initialize	This action calls protected method resetVariables (), in this method all messages (error message and other) and tag reads are cleared. Associated timers will be canceled.
Play Audio	This action plays the audio set by calling the method playAudio ()
Post Application Event	This action loops through the scanned tag list and gets the tag id information, loops through the scanned scale list and gets the weight information, and gets the all the event parameter values, using these it forms the string and post the application event with name of event configured with event type as “PROCESS”.
Post Application Event for single tag	This action post the event with name configured in edge process properties for single tag by forming the string using the event parameters and with event type as “PROCESS”
Post Process Fail Event	This action set status of corresponding **EdgeProcessRuntime** as fail and calls related edge process **processStatus** method with parameter as integer constant which represent as fail
Post Process Status Event	**This action** calls related edge process **processStatus** method with status of **EdgeProcessRuntime** as it’s parameter
Post Process Success Event	This action set status of corresponding **EdgeProcessRuntime** as success and calls related edge process **processStatus** method with parameter as integer constant which represent as success
Post Tag Read Event	This action post **Tag Read** event, with event type as **DEVICEREAD** for the first entry in the scanned tag list and clears the scanned tag list.
Process PLC Read	Method corresponding to this action is

	empty. (No code inside the method)
Process Scale Read	Method corresponding to this action is empty. (No code inside the method)
Process Tag Read	Method corresponding to this action is empty. (No code inside the method)
Remove All Event Parameters	This action removes all the event parameters
Remove All Global Parameters	This action removes all the parameters which are used across the **EdgeProcessRuntime** objects
Remove All Parameters	Removes all entries from the _parameters in the parent EdgeProcessCO. (Note: Not able to get what type of information stored in this map)
Reset	This action reset all the variables of the control object, stops the reader timer, delay wait timer by calling "**Stop Read Timer**", "**Stop Delay Wait Timer**" respectively and clears the read history. Finally **resetCompleted** event is raised.
Reset Delay Wait Timer	This action resets the **Delay Timer** by calling the **Stop Delay Wait Timer** and **Start Delay Wait Timer**
Reset Read Timer	This action resets the **Read Timer** by calling the **Stop Read Timer** and **Start Read Timer**
Retrieve Context List from Tracking	This action gets the item details for each tag id from the Tracking Manager. (**As EOL for Tracking Manager this action should be removed**)
Start Delay Wait Timer	This action starts time for the **Delay Timer**
Start Read timer	This action starts time for the **Read Timer**
Start Reading	This action sets boolean **_startRead** to **true**
Start reset Timer	This action starts time for the **Reset Timer** and status of the **EdgeProcessRunTime** set as none
Start Delay wait timer	This action stops the **Delay Timer**
Stop read timer	This action stops the **Read Timer**
Stop reading	This action sets boolean **_startRead** to **false**
Stop reset timer	This action stops the **Reset Timer**
Transition to init state	This action raises **toInitState** event through corresponding edge process
Transition to next state	This action raises **toNextState** event through corresponding edge process
Validate PLC read	Method corresponding to this action is empty. (No code inside the method)
Validate scale read	Method corresponding to this action is empty. (No code inside the method)

Validate single Tag read	This action gets the first entry from the scanned tag list which is not processed by checking for the **processed time** (if process time is zero then it is not processed) and validates the tag by calling the **validateTag** method passing the tag id. If it returns true then "**Post Process Success Event**" action will be called otherwise "**Post Process Fail Event**" action will be called
Validate tag reads	This action loops through the scanned tag list and for each tag is validated against the tag validations configured if scanned tag id is failed for one tag validation then it will be removed from the scanned tag list and added to the **_issueTagList** map (key as tag and DataRead as value)

Script Methods

Script methods are those which can be used in the PreScript and PostScript writing while configuring the EdgeProcess.

- init (): This method first put the put the event parameters with event field and values and calls the super class EdgeProcessCO.init () method.

Note: Apart from these we can use the super class EdgeProcessCO script methods also

6.3.3 EdgeProcessCO

This control object is the Base class for all other control objects. Whenever user wants to build his/her own control object, it should extend this and build on top of this. This EdgeProcessCO control object provides almost all basic event actions except very specific business event actions.

Actions: The following actions can be selected from the UI. Scripts cannot use these actions. All these actions take StateMachineContext as an input parameter.

Action	Description
Add Tags to UI	This action puts the scanned parent tag id and child tag ids into the list in the form of ReadHistory object. Before putting it checks the whether already ReadHistory with parent tag id or scanned tag ids exists. If so it won't add to the list otherwise it adds to the list. (But note that in this action there is no setting of parent tag in the ReadHistory object is done)
Clear Scanned Tags List	This action clears all tags scanned from the scanned tag list
Collect Unique Reads	This action loop through the all the scanned tag list, checks whether tag read time is set or not, if not then sets the tag read time and put the tag id as key and DataRead object as value in the hash map. After putting into the

	hash map it removes that entry from the scanned tag list
Disable the device	This action **disable** all the devices associated to the corresponding edge process
Enable The Device	This action **enable** all the devices associated to the corresponding edge process
Extend the timer for unique tag	This action loops through the scanned tag list and check for each tag detail is there in the unique tag list or not. If it is not there then it “Reset Delay Wait Timer” action is called. And if tag read time is not set then it sets tag read time
Initialize	This action calls protected method resetVariables (), in this method all messages (error message and other) and tag reads are cleared. Associated timers will be canceled.
Play Audio	This action plays the audio set by calling the method playAudio ()
Post Application Event	This action loops through the scanned tag list and gets the tag id information, loops through the scanned scale list and gets the weight information, and gets the all the event parameter values, using these it forms the string and post the application event with name of event configured with event type as “PROCESS”.
Post Application Event For Single Tag	This action post the event with name configured in edge process properties for single tag by forming the string using the event parameters and with event type as “PROCESS”
Post Process Fail Event	This action set status of corresponding **EdgeProcessRuntime** as fail and calls related edge process **processStatus** method with parameter as integer constant which represent as fail
Post Process Status Event	**This action** calls related edge process **processStatus** method with status of **EdgeProcessRuntime** as it’s parameter
Post Process Success Event	This action set status of corresponding **EdgeProcessRuntime** as success and calls related edge process **processStatus** method with parameter as integer constant which represent as success
Post Tag Read Event	This action post **Tag Read** event, with event type as **DEVICEREAD** for the first entry in the scanned tag list and clears the scanned tag list.
Process PLC Read	Method corresponding to this action is empty. (No code inside the method)

Process Scale Read	Method corresponding to this action is empty. (No code inside the method)
Process Tag Read	Method corresponding to this action is empty. (No code inside the method)
Remove All Event Parameters	This action removes all the event parameters
Remove All Global Parameters	This action removes all the parameters which are used across the **EdgeProcessRuntime** objects
Remove All Parameters	Removes all entries from the _parameters in the parent EdgeProcessCO. (Note: Not able to get what type of information stored in this map)
Reset	This action reset all the variables of the control object, stops the reader timer, delay wait timer by calling "**Stop Read Timer**", "**Stop Delay Wait Timer**" respectively and clears the read history. Finally **resetCompleted** event is raised.
Reset Delay Wait Timer	This action resets the **Delay Timer** by calling the **Stop Delay Wait Timer** and **Start Delay Wait Timer**
Reset Read Timer	This action resets the **Read Timer** by calling the **Stop Read Timer** and **Start Read Timer**
Retrieve Context List from Tracking	This action gets the item details for each tag id from the Tracking Manager. (**As EOL for Tracking Manager this action should be removed**)
Start Delay Wait Timer	This action starts time for the **Delay Timer**
Start Read Timer	This action starts time for the **Read Timer**
Start Reading	This action sets boolean **_startRead** to **true**
Start Reset Timer	This action starts time for the **Reset Timer** and status of the **EdgeProcessRunTime** set as none
Stop Delay Wait Timer	This action stops the **Delay Timer**
Stop Read Timer	This action stops the **Read Timer**
Stop Reading	This action sets boolean **_startRead** to **false**
Stop Reset Timer	This action stops the **Reset Timer**
Transition to Init State	This action raises **toInitState** event through corresponding edge process
Transition to Next State	This action raises **toNextState** event through corresponding edge process
Validate PLC Read	Method corresponding to this action is empty. (No code inside the method)
Validate Scale Read	Method corresponding to this action is empty. (No code inside the method)
Validate Single Tag Read	This action gets the first entry from the scanned tag list which is not processed by

	checking for the **processed time** (if process time is zero then it is not processed) and validates the tag by calling the **validateTag** method passing the tag id. If it returns true then "**Post Process Success Event**" action will be called otherwise "**Post Process Fail Event**" action will be called
Validate Tag Reads	This action loops through the scanned tag list and for each tag is validated against the tag validations configured if scanned tag id is failed for one tag validation then it will be removed from the scanned tag list and added to the **_issueTagList** map (key as tag and DataRead as value)

Script Methods

Script methods are those which can be used in the PreScript and PostScript writing while configuring the EdgeProcess.

- init () : It initializes the various attributes of this control object and also adds the various tag validation encoders configured for this control object.
- playAudio (): This method plays an audio of buzzer kind of thing using the configured audio
- validateTag (String tag): This method validate the input tag against the tag validations and checks if it is a child tag, if so then returns true otherwise returns false.
- setGlobalParameter (String name, Object value): This method creates the entry in the map with key as name and value as passed object. This map is available to all the control objects so that any edge process can access.
- removeGlobalParameter (String name): This method removes the entry from the map which is having this name as key
- getGlobalParameters (): This method returns the global parameter map which is holding the name of the parameters and its corresponding values.
- removeAllGlobalParameter (): This method removes all the entries from the global parameter map
- setParameter (String name, String value): This method put the entry into the EdgeProcessCO map which holds the parameter name and its value.
- getParameter (String name) : This method gets the value from the EdgeProcessCO map which is having key as the passed name
- removeParameter (String name): This method removes the value from the EdgeProcessCO parameter map which is having key as the passed name.
- getAllParameter (): This method returns the EdgeProcessCO parameters map
- removeAllParameter (): This method removes all the entries from the EdgeProcessCO parameters map
- setEventParameter (String name, String value): This method creates entry into the event parameter map with key as name and value as the passed parameter value.
- getEventParameter (String name): This method gets the value from the EdgeProcessCO event parameter map which is having key as the passed name.
- removeEventParameter (String name): This method removes the entry from EdgeProcessCO event parameter map which is having key as the passed name.

- removeAllEventParameter (): This method removes all the entries from the EdgeProcessCO event parameters map.
- getEventParameters (): This method returns the EdgeProcessCO event parameters map.
- getIssueTagList (): This method returns issue tag list
- postProcessEvent (String event): This method takes the event name and calls the EdgeProcessEP.sendEvent method by passing the parameter.
- resetVariables (): This method set the status as none, clear the message, error message, time message, reads, and stop all the timers
- stopTimers (): Cancels all the associated timers
- startTimer (String timerName, int timeoutPeriod, String event): This method starts the timer with the passed timerName and sets the time out as passed timeoutPeriod. “Timer Started” event is raised. After the specified time is over (time out period) then event will be raised with the event name as passed parameter “event”.
- stopTimer (String timerName, String eventName): This method cancels the timer with name as passed parameter timerName and raises the event with name as passed parameter eventName.
- clearReads (): This method clears all the lists which is holding tag reads
- getContextList (): This method gets the map containing the tag id and its details. (This actually calling Tracking Manager to get the details which is not used anymore)
- getParentTagList (): This method gets the parent tag list.
- checkForDuplication (ReadHistory read): This method loops through the list which is holding the ReadHistory objects and checks if passed ReadHistory object is there or not by comparing parent read and other reads are equal or not. If found then returns true otherwise false.
- getHistory (): This method returns the list of ReadHistory objects
- isReadStart (): It returns the boolean value, if it returns false then tag read is not yet started and if returns true then reading of tags is started.
- getStatus (): This method returns the integer which is represents the EdgeProcessRunTime status.
- getTimerMessage (): This method returns the timer message
- getScannedTagList (): This method returns the map containing the read tags and it’s details
- getScannedTagCount (): This method returns the size of the map containing read tags and it’s details
- getScannedPLCList ():This method returns the map containing the PLC device read tags and it’s details
- getScannedPLCCount ():This method returns size of map containing the PLC device read tags and it’s details
- getScannedScaleList ():This method returns the map containing the Scale device read tags and it’s details
- getScannedScaleCount ():This method returns size of map containing the Scale device read tags and it’s details
- onDataEvent (String deviceName, String location, LinkedList<String[]> data): This method depending on the device it will raise an event using the location and data which is holding the various tag read details.

- addToScannedTagList (String device, String location, LinkedList<String[]> data): This method loops through passed data list and add the details to the map holding the tag read and its details
- addToScannedPLCList (String device, String location, LinkedList<String[]> data): This method loops through passed data list and add the details to the map holding the PLC device tag read and its details
- postApplicationEvent (String applicationEventName): This sets the application event name attribute of this object to the passed applicationEventName parameter. And calls the postApplicationEvent (StateMachineContext).
- addToScannedScaleList (String device, String location, LinkedList<String[]> data): This method loops through passed data list and add the details to the map holding the Scale device tag read and its details
- postProcessEvent (String event, String device, String location, LinkedList<String[]> data): This sets the all the passed parameter to the StateMachineContext and calls the event provider sendEvent (String, StateMachineContext) method.
- getMessageHeader (): This method returns EdgeProcessRunTime status along with zone, device, organization, and time.
- getMessageHeader (DataRead read): This method is same as the getMessageHeader () method but the location and time is fetched from the passed parameter DataRead object instead of EdgeProcessCO object instance.
- getApplicationEventName (): Returns the application event name of the EdgeProcessCO object instance
- setApplicationEventName (String event): Sets the application event name as the passed parameter event
- getLocation (): Returns the location.
- setLocation (String location): Sets the location with the passed parameter.
- setAudio (String audio): Sets the audio
- getDocumentField (): Returns the value of the Document Field parameter.
- setDocumentField (String documentField): Sets the value for Document Field parameter.
- getDocumentFieldValue (): Returns the value of the Document Field Value parameter.
- setDocumentFieldValue (String documentFieldValue): Sets the value for the Document Field Value parameter.
- getDocumentList (): Returns the map containing the document number and its details. (Remote call to Sensornet to get the details).
- reset (): Calls the method reset(StateMachineContext context) to reset all the variables, timers, and messages associated to edge process
- postEvent (String eventName, String eventType, String content): This method post the event with name eventName, event type as eventType with data in it as content.
- logDebug (String msg): This method logs the message into the edgeware debug log file as DEBUG statement
- logInfo (String msg): This method logs the message into the edgeware debug log file as INFO statement

6.3.4 MovementProcessCO

This control object is used to raise events for the Route Management by getting the scanned items' current location and comparing this with the location configured in parent EdgeProcessCO control

object. If items' location is null or it is not equal to the location configured in parent EdgeProcessCO then it means items are entering into the configured location and those will be added to the list which is called as _enterList and if scanned items' location is same as the configured location then that means items are leaving from that location and those will be added to _leaveList.

Depending upon the enter list and leave list this control object raised different events.

Getting information for items in this control object is done by remotely calling the Tracking Manager. As end of life for Tracking Manager, either we need to remove this control object or else modify the control object to hit the EdgeBO tables.

Actions: The following actions can be selected from the UI. Scripts cannot use these actions. All these actions take StateMachineContext as an input parameter.

Actions	Description
Add Tags to UI	This action puts the scanned parent tag id and child tag ids into the list in the form of ReadHistory object. Before putting it checks the whether already ReadHistory with parent tag id or scanned tag ids exists. If so it won't add to the list otherwise it adds to the list. (But note that in this action there is no setting of parent tag in the ReadHistory object is done)
Clear Scanned Tags List	This action clears all tags scanned from the scanned tag list
Collect Unique Reads	This action loop through the all the scanned tag list, checks whether tag read time is set or not, if not then sets the tag read time and put the tag id as key and DataRead object as value in the hash map. After putting into the hash map it removes that entry from the scanned tag list
Disable the device	This action **disable** all the devices associated to the corresponding edge process
Enable the device	This action **enable** all the devices associated to the corresponding edge process
Extend the timer for unique tag	This action loops through the scanned tag list and check for each tag detail is there in the unique tag list or not. If it is not there then it "Reset Delay Wait Timer" action is called. And if tag read time is not set then it sets tag read time
Initialize	This action calls protected method resetVariables (), in this method all messages (error message and other) and tag reads are cleared. Associated timers will be canceled.
Play Audio	This action plays the audio set by calling the method playAudio ()
Post Application Event	This action loops through the scanned tag list and gets the tag id information, loops through the scanned scale list and gets the weight information, and gets the all the event parameter values, using these it forms the string and post the application event with name of event configured with event type as

	“PROCESS”.
Post Application Event for single tag	This action post the event with name configured in edge process properties for single tag by forming the string using the event parameters and with event type as “PROCESS”
Post Enter Event	This action raises the configured enter event name with event type “PROCESS” using all the tag id of the enter list. In this event Next Location is set as configured _enterNextLocation
Post Enter Event from Item’s Next Location	This action gets the Next Location from one of the item in the enter list and if it is not null then configured location is set as the Next Location else configured location is set as the Location of the item. In the raised event location is set as the this changed configured location (EdgeProcessCO.getMessageHeader()) and Next Location is _enterNextLocation After this it raises the configured enter event name with event type “PROCESS” using all the tag id of the enter list.
Post Enter Event from Next location	This action in the event location is set as the configured _enterNextLocation and Next Location is set as _enterNextLocation and raises the configured enter event name with event type “PROCESS” using all the tag id of the enter list.
Post Leave Event from Enter Next Location	This action in the event location is set as the configured _enterNextLocation and Next Location as _leaveNextLocation raises the configured leave event name with event type “PROCESS” using all the tag id of the leave list.
Post Leave Event from Next location	This action in the event location and Next Location is set as the configured _leaveNextLocation and raises the configured leave event name with event type “PROCESS” using all the tag id of the leave list.
Post Leave Event	This action raises the configured leave event name with event type “PROCESS” using all the tag id of the leave list. In this event Next Location is set as configured _leaveNextLocation
Post Process Fail Event	This action set status of corresponding **EdgeProcessRuntime** as fail and calls related edge process **processStatus** method with parameter as integer constant which represent as fail
Post Process Status Event	**This action** calls related edge process **processStatus** method with status of **EdgeProcessRuntime** as it’s parameter
Post Process Success Event	This action set status of corresponding

	EdgeProcessRuntime as success and calls related edge process **processStatus** method with parameter as integer constant which represent as success
Post Tag Read Event	This action post **Tag Read** event, with event type as **DEVICEREAD** for the first entry in the scanned tag list and clears the scanned tag list.
Process PLC Read	Method corresponding to this action is empty. (No code inside the method)
Process Scale Read	Method corresponding to this action is empty. (No code inside the method)
Process TagRead	This action for each tag read get the Location and if it is not null and equals to the location configured then it will add it to the leave list otherwise it will be added to the enter list
Remove All Event Parameters	This action removes all the event parameters
Remove All Global Parameters	This action removes all the parameters which are used across the **EdgeProcessRuntime** objects
Remove All Parameters	Removes all entries from the _parameters in the parent EdgeProcessCO. (Note: Not able to get what type of information stored in this map)
Reset	This action reset all the variables of the control object, stops the reader timer, delay wait timer by calling "**Stop Read Timer**", "**Stop Delay Wait Timer**" respectively and clears the read history. Finally **resetCompleted** event is raised.
Reset Context List	This action clears the map which holds information of items' got from the tracking manager
Reset Delay Wait Timer	This action resets the **Delay Timer** by calling the **Stop Delay Wait Timer** and **Start Delay Wait Timer**
Reset Enter List	This action clears set which stores the enter reads.
Reset Leave List	This action clears set which stores the leaving reads.
Reset Read Timer	
Retrieve Context List from Tracking Manager	This action gets the all the scanned tag list information from the Tracking Manager Database through remote call using SOAP.
Start Delay Wait Timer	This action starts time for the **Delay Timer**
Start Read timer	This action starts time for the **Read Timer**
Start Reading	This action sets boolean **_startRead** to **true**
Start reset Timer	This action starts time for the **Reset Timer** and status of the **EdgeProcessRunTime** set as none

Start Delay wait timer	This action stops the **Delay Timer**
Stop read timer	This action stops the **Read Timer**
Stop reading	This action sets boolean **_startRead** to **false**
Stop reset timer	This action stops the **Reset Timer**
Transition to Init State	This action raises **toInitState** event through corresponding edge process
Transition to Next State	This action raises **toNextState** event through corresponding edge process
Validate PLC read	Method corresponding to this action is empty. (No code inside the method)
Validate scale read	Method corresponding to this action is empty. (No code inside the method)
Validate single Tag read	This action gets the first entry from the scanned tag list which is not processed by checking for the **processed time** (if process time is zero then it is not processed) and validates the tag by calling the **validateTag** method passing the tag id. If it returns true then "**Post Process Success Event**" action will be called otherwise "**Post Process Fail Event**" action will be called
Validate tag reads	This action loops through the scanned tag list and for each tag is validated against the tag validations configured if scanned tag id is failed for one tag validation then it will be removed from the scanned tag list and added to the **_issueTagList** map (key as tag and DataRead as value)

Script Methods

Script methods are those which can be used in the PreScript and PostScript writing while configuring the EdgeProcess.

init (): This method calls the EdgeProcessCO.init () method and assigns the value to enter event name, leave event name, enter action, leave action, enter next location, and leave next location of this class.

6.3.5 ReceivingProcessCO

This control object takes the scanned list and gets the details of the items scanned through remote call to Sensornet. (**Note: The method it is calling getExpectedTrackableItems in the class com.insync.iApp.sensornet.soap.ServiceExecution is not there. Need to modify the remote call according to the purpose of this control object**). It gets what are the expected count, scanned count, matched count, and issue count (don't know what this count means) and raises events using these count values.

Actions: The following actions can be selected from the UI. Scripts cannot use these actions. All these actions take StateMachineContext as an input parameter.

Action	Description
Add Tags to UI	This action puts the scanned parent tag id and child tag ids into the list in the form of ReadHistory object. Before putting it checks the whether already ReadHistory with parent tag id or scanned tag ids exists. If so it won't add to the list otherwise it adds to the list. (But note that in this action there is no setting of parent tag in the ReadHistory object is done)
Clear Scanned Tags List	This action clears all tags scanned from the scanned tag list
Collect Unique Reads	This action loop through the all the scanned tag list, checks whether tag read time is set or not, if not then sets the tag read time and put the tag id as key and DataRead object as value in the hash map. After putting into the hash map it removes that entry from the scanned tag list
Disable the device	This action **disable** all the devices associated to the corresponding edge process
Enable the device	This action **enable** all the devices associated to the corresponding edge process
Extend the timer for unique tag	This action loops through the scanned tag list and check for each tag detail is there in the unique tag list or not. If it is not there then it "Reset Delay Wait Timer" action is called. And if tag read time is not set then it sets tag read time
Get Expected List	This action gets the scanned tag id details from the Sensornet through remote call using SOAP and details will be put into the RemoteCall object map. And data reads formed from by calling the Sensornet through remote call will be added to the issue tag list.
Initialize	This action calls protected method resetVariables (), in this method all messages (error message and other) and tag reads are cleared. Associated timers will be canceled.
Match Expected Reads	In this action 1. Checks if there is no records in the map of RemoteCall object (Not able to find records in the Sensornet for scanned tags) then Fail event will be raised 2. Checks if there is no parent tag in the RemoteCall object then Fail event will be raised. 3. Gets all the children for a parent from the RemoteCall object map. It loops through this

	list and with the help of issueTagList, scanned tag list it calculate the matching count (how many scanned reads are found in the database for the parent). If the matching count is not equal to the size of the children got from RemoteCall object map then it is partial receive and post the Fail event. If it is equal then it will post Success event.
Play Audio	This action plays the audio set by calling the method playAudio ()
Post Application Event	This action post the configured application event name with the details like expected count, match count, issue count, scanned count, and processed tag list.
Post Application Event for single tag	This action post the event with name configured in edge process properties for single tag by forming the string using the event parameters and with event type as “PROCESS”
Post Issue Event	This action posts the configured issue event with the details like expected count, match count, issue count, scanned count, and issue tag list.
Post Process Fail Event	This action set status of corresponding **EdgeProcessRuntime** as fail and calls related edge process **processStatus** method with parameter as integer constant which represent as fail
Post Process Status Event	**This action** calls related edge process **processStatus** method with status of **EdgeProcessRuntime** as it’s parameter
Post Process Success Event	This action set status of corresponding **EdgeProcessRuntime** as success and calls related edge process **processStatus** method with parameter as integer constant which represent as success
Post Tag Read Event	This action post **Tag Read** event, with event type as **DEVICEREAD** for the first entry in the scanned tag list and clears the scanned tag list.
Post PLC Read	Method corresponding to this action is empty. (No code inside the method)
Process Scale Read	Method corresponding to this action is empty. (No code inside the method)
Process Tag Read	Method corresponding to this action is empty. (No code inside the method)
Remove All Event Parameters	This action removes all the event parameters
Remove All Global Parameters	This action removes all the parameters which are used across the **EdgeProcessRuntime** objects

Remove All Parameters	Removes all entries from the _parameters in the parent EdgeProcessCO. (Note: Not able to get what type of information stored in this map)
Reset	This action reset all the variables of the control object, stops the reader timer, delay wait timer by calling "**Stop Read Timer**", "**Stop Delay Wait Timer**" respectively and clears the read history. Finally **resetCompleted** event is raised.
Reset Delay Wait Timer	This action resets the **Delay Timer** by calling the **Stop Delay Wait Timer** and **Start Delay Wait Timer**
Reset Read Timer	This action resets the **Read Timer** by calling the **Stop Read Timer** and **Start Read Timer**
Retrieve Context List from Tracking	This action gets the item details for each tag id from the Tracking Manager. (**As EOL for Tracking Manager this action should be removed**)
Start Delay Wait Timer	This action starts time for the **Delay Timer**
Start Package	This action calls the startPackage () method and in this method package number will be generated by the current time GMT milliseconds and put the event details by calling getMessageHeader () and post the event "Start Receiving" with event type as "PROCESS"
Start Read Timer	This action starts time for the **Read Timer**
Start Reading	This action sets boolean **_startRead** to **true**
Start Reset Timer	This action starts time for the **Reset Timer** and status of the **EdgeProcessRunTime** set as none
Stop Delay Wait Timer	This action stops the **Delay Timer**
Stop package	This action calls the stopPackage () method and in this method package number is set as null and put the event details by calling getMessageHeader () and post the event "Stop Receiving" with event type as "PROCESS"
Stop read timer	This action stops the **Read Timer**
Stop reading	This action sets boolean **_startRead** to **false**
Stop reset timer	This action stops the **Reset Timer**
Transition to init state	This action raises **toInitState** event through corresponding edge process
Transition to next state	This action raises **toNextState** event through corresponding edge process
Validate PLC read	Method corresponding to this action is empty. (No code inside the method)

Validate scale read	Method corresponding to this action is empty. (No code inside the method)
Validate single Tag read	This action gets the first entry from the scanned tag list which is not processed by checking for the **processed time** (if process time is zero then it is not processed) and validates the tag by calling the **validateTag** method passing the tag id. If it returns true then "**Post Process Success Event**" action will be called otherwise "**Post Process Fail Event**" action will be called
Validate tag read	This action loops through scanned tag list first checks for is there any parent tag is there through configured Tag Validations, if it is then it will be added to the parent tag list of RemoteCall object. And also put into the control object process tag list. If it is not parent tag id then it checks for is it a child tag id through configured Tag Validations, if it is then it is put into the control object process tag list. After this it removes all the entries from the scanned tag list which are put into the process tag list. Remaining entries in the scanned tag list are put into the issue tag list

Script Methods

Script methods are those which can be used in the PreScript and PostScript writing while configuring the EdgeProcess.

- init (): This method calls the EdgeProcessCO.init () method and assigns the values for type of item, parent tag encoder, child tag encoder, to the remote call object sets the what's the parent tag field in the Sensornet and sets the issue event name. Finally calls the resetVariables () method.
- getParentTag (): This method returns the first parent tag from the parent tag list from the RemoteContextCall object
- getParentTagList (): This method returns the list of parent tags from the RemoteContextCall object
- getParentTagCount (): Returns size of parent tag list from the RemoteContextCall object
- getExpectedCount (): This method gets count of all the children of all the parent tags.
- getMatchedCount (): Returns process tag list size.
- getIssueCount (): Returns the issue tag list size
- getTotalScanCount (): It loops through the issue tag list and returns the list count which is not having process time is not zero.
- resetVariables (): Calls EdgeProcessCO.resetVariables () method and calls RemoteContextCall.clearAll () method.
- startPackage (): In this method if package number is null or empty string it gets the current milliseconds to the package number and calls the super class EdgeProcessCO.getMessageHeader() method. Append the package number information

to the header and post the event with name Start Receiving, event type as PROCESS with data as header with package number.

- stopPackage (): Sets the package number as null and calls the super class EdgeProcessCO.getMessageHeader () method. Append the package number information to the header and post the event with name Stop Receiving, event type as PROCESS with data as header with package number.
- isPackageStart (): Returns boolean value true if package number is not null else returns false.
- holdReceiving (String username): This method first calls the super class EdgeProcessCO.getMessageHeader () to this header message adds information like type of item, parent field with each parent tag details and also with the issue tag list ids and post the event with name Hold, event type as PROCESS with data as described in previous sentence.

6.3.6 SafetyEdgeProcessCO

This control object used to send the device command for device type PLC depending upon the position of the scanned tags.

Actions: The following actions can be selected from the UI. Scripts cannot use these actions. All these actions take StateMachineContext as an input parameter.

Action	Description
Add Tags to UI	This action puts the scanned parent tag id and child tag ids into the list in the form of ReadHistory object. Before putting it checks the whether already ReadHistory with parent tag id or scanned tag ids exists. If so it won't add to the list otherwise it adds to the list. (But note that in this action there is no setting of parent tag in the ReadHistory object is done)
Clear Scanned Tags List	This action clears all tags scanned from the scanned tag list
Collect Unique Reads	This action loop through the all the scanned tag list, checks whether tag read time is set or not, if not then sets the tag read time and put the tag id as key and DataRead object as value in the hash map. After putting into the hash map it removes that entry from the scanned tag list
Disable the device	This action **disable** all the devices associated to the corresponding edge process
Enable the device	This action **enable** all the devices associated to the corresponding edge process
Extend the timer for unique tag	This action loops through the scanned tag list and check for each tag detail is there in the unique tag list or not. If it is not there then it "Reset Delay Wait Timer" action is

	called. And if tag read time is not set then it sets tag read time
Initialize	This action calls protected method resetVariables (), in this method all messages (error message and other) and tag reads are cleared. Associated timers will be canceled.
Play Audio	This action plays the audio set by calling the method playAudio ()
Post Application Event	This action loops through the scanned tag list and gets the tag id information, loops through the scanned scale list and gets the weight information, and gets the all the event parameter values, using these it forms the string and post the application event with name of event configured with event type as “PROCESS”.
Post Application Event for single tag	This action post the event with name configured in edge process properties for single tag by forming the string using the event parameters and with event type as “PROCESS”
Post Process Fail Event	This action set status of corresponding **EdgeProcessRuntime** as fail and calls related edge process **processStatus** method with parameter as integer constant which represent as fail
Post Process Status Event	**This action** calls related edge process **processStatus** method with status of **EdgeProcessRuntime** as it’s parameter
Post Process Success Event	This action set status of corresponding **EdgeProcessRuntime** as success and calls related edge process **processStatus** method with parameter as integer constant which represent as success
Post Tag Read Event	This action gets the details like Device, Location, Organization, Time, status, Tag Id, Tag Read Time, PosX, and PosY for the first read and post the event “Tag Read” with event type as “DeviceRead”
Process PLC Read	Method corresponding to this action is empty. (No code inside the method)
Process Scale Read	Method corresponding to this action is empty. (No code inside the method)
Process Tag Read	Method corresponding to this action is empty. (No code inside the method)
Remove All Event Parameters	This action removes all the event parameters
Remove All Global Parameters	This action removes all the parameters which are used across the **EdgeProcessRuntime** objects
Remove All Parameters	Removes all entries from the _parameters in

	the parent EdgeProcessCO. (Note: Not able to get what type of information stored in this map)
Reset	This action reset all the variables of the control object, stops the reader timer, delay wait timer by calling "**Stop Read Timer**", "**Stop Delay Wait Timer**" respectively and clears the read history. Finally **resetCompleted** event is raised.
Reset Delay Wait Timer	This action resets the **Delay Timer** by calling the **Stop Delay Wait Timer** and **Start Delay Wait Timer**
Reset Read Timer	This action resets the **Read Timer** by calling the **Stop Read Timer** and **Start Read Timer**
Retrieve Context List from Tracking	This action gets the item details for each tag id from the Tracking Manager. (**As EOL for Tracking Manager this action should be removed**)
Start Delay Wait Timer	This action starts time for the **Delay Timer**
Start Read timer	This action starts time for the **Read Timer**
Start Reading	This action sets boolean **_startRead** to **true**
Start reset Timer	This action starts time for the **Reset Timer** and status of the **EdgeProcessRunTime** set as none
Stop Delay wait timer	This action stops the **Delay Timer**
Stop read timer	This action stops the **Read Timer**
Stop reading	This action sets boolean **_startRead** to **false**
Stop reset timer	This action stops the **Reset Timer**
Transition to init state	This action raises **toInitState** event through corresponding edge process
Transition to next state	This action raises **toNextState** event through corresponding edge process
Turn green off	This action loops through all devices associated to the edge process and checks for each device for type PLC, if it so then calls the **sendCommand** on that device by passing the configured **Green Off Command** value
Turn green on	This action loops through all devices associated to the edge process and checks for each device for type PLC, if it so then calls the **sendCommand** on that device by passing the configured **Green On Command** value
Turn red off	This action loops through all devices associated to the edge process and checks for each device for type PLC, if it so then calls the **sendCommand** on that device by passing the configured **Red Off Command**

	value
Turn red on	This action loops through all devices associated to the edge process and checks for each device for type PLC, if it so then calls the **sendCommand** on that device by passing the configured **Red On Command** value
Validate PLC read	Method corresponding to this action is empty. (No code inside the method)
Validate scale read	Method corresponding to this action is empty. (No code inside the method)
Validate single Tag read	This action gets the first entry from the scanned tag list which is not processed by checking for the **processed time** (if process time is zero then it is not processed) and validates the tag by calling the **validateTag** method passing the tag id. If it returns true then "**Post Process Success Event**" action will be called otherwise "**Post Process Fail Event**" action will be called
Validate tag read	This action loops through the scanned reads and for each read it gets the PosX parameter value and PosY parameter value. It checks PosX parameter value is between the configured start value of x and end value of x, similarly for PosY parameter value. If it is not then _greenLight variable is set to Red if it is not already red else it will be set to Green if it not already green. After setting green it will call postProcessSuccess method.

Script Methods

Script methods are those which can be used in the PreScript and PostScript writing while configuring the EdgeProcess.

- init (): This method sets the value for all the variables and calls the EdgeProcessCO.init () method.

6.3.7 PLCDeviceCO

Actions: The following actions can be selected from the UI. Scripts cannot use these actions. All these actions take StateMachineContext as an input parameter. To control a PLC device, you have to configure all pins controlled by the device and the "on" value. "on" value could be 1 or 0.

Initialize	Implementation of abstract method. Do nothing.
Disable	Close connection to the device.
Enable	Open connection to the device.
Send Command	Send command to the device. The event context must contains

	“command” parameter with the command value to be sent.
Turn On All Output Signals	Set all output signals to “on” value.
Turn Off All Output Signals	Set all output signals to “off” value.
Turn Orange Light	Set orange pin to “on” value.
Blink Red Light	Set red blink pin to “on” value.
Blink Green Light	Set green blink pin to “on” value.
Blink Orange Light	Set orange blink pin to “on” value.
Turn Green Light	Set green pin to “on” value.
Turn Red Light	Set red pin to “on” value.
Clear All Light	Set all light pins to “off” value.
Clear Eyes	Set all pins to “off” value.
Set Single Eye	Set single eye flag to indicate the device only controls one eye.

Script Methods

Script methods are those which can be used in the PreScript and PostScript writing while configuring the EdgeProcess. Assume the device name is “EntryEye”, a sample script could be:

```
EntryEye_CO.setEventProvider();
```

- setEventProvider (): this method is to set "PLCDeviceEP" as event provider.
- init (): this method associates device with this controlled object and reads all configured properties.
- readInputSignal (int pin): this method reads signal value on provided pin.
- readAllInputSignals (): this method reads signal values of all input pins.
- sendCommand (String command): this method send provided command to the device.
- isSingleEye (): return true if the device only controls one eye.
- getEyePosition (int eye): return value indicates if provided eye is for entry or exit.
- getDirection (): return current moving direction: UNKNOWN, IN or OUT.
- setEnterEye (boolean value): this method sets direction to OUT if value is true and previous direction is unknown.
- setExitEye (boolean value): this method sets direction to IN if value is true and previous direction is unknown.
- isEntering (): return true after seeing exit eye signal before seeing enter eye signal.
- isEntered (): return true after seeing both eye signals and exit signal is received before enter eye signal.
- isExiting (): return true after seeing enter eye signal before seeing exit eye signal.
- isExited (): return true after seeing both signals and enter signal is received before exit signal.
- getEventProvider (): return EventProvider object.
- setEventProvider (PLCDeviceEP eventProvider): sets EventProvider object for this edge process.

6.3.8 PrinterDeviceCO

Actions: The following actions can be selected from the UI. Scripts cannot use these actions. All these actions take StateMachineContext as an input parameter.

Initialize	Gets printer from configuration and associate it with this controlled object.
Print	Sends command to the printer. setCommand method must called before this action.

Script Methods

Script methods are those which can be used in the PreScript and PostScript writing while configuring the EdgeProcess.

- getPrinterName (): returns printer name configured in SN.
- print (String data): Send provided data to printer.
- setCommand (String command): set local variable with provided “command” for Print action.

6.3.9 ReaderDeviceCO

Actions: The following actions can be selected from the UI. Scripts cannot use these actions. All these actions take StateMachineContext as an input parameter.

Initialize	Implementation of abstract method. Do nothing.
Disable Device Read	Close connection to the device.
Enable Device Read	Open connection to the device.

Script Methods

Script methods are those which can be used in the PreScript and PostScript writing while configuring the EdgeProcess.

- init (): Associate configured device to this controlled object.
- sendCommand (String command): send supplied command to the device.
- writeID (String tagID, String antennaID): send write tag command to the device with supplied tagID and antennaID.
- killTag (String tagID, String password, String antennaID): send kill tag command to the device with supplied tagID and antennaID.
- lockTag (String tagID, String password, String antennaID): send lock tag command to the device with supplied parameters.
- readMemoryBank (String tagID, int bankID, int blockID, int blockCount, String antennaID): send read memory bank command with supplied parameters and return the response from the device.
- writeMemoryBank (String tagID, int bankID, int blockID, String hexData, String antennaID) send write memory bank command to the device.

6.3.10 ScaleDeviceCO

Actions: The following actions can be selected from the UI. Scripts cannot use these actions. All these actions take StateMachineContext as an input parameter.

Initialize	Implementation of abstract method. Do nothing.
Stop Scale Read	Close connection to the device.
Start Scale Read	Open connection to the device.

Script Methods

Script methods are those which can be used in the PreScript and PostScript writing while configuring the EdgeProcess.

- init (): Associates configured device with this controlled object.
- setEventProvider (): set ScaleDeviceEP as event provider for this controlled object.
- sendCommand (String command): send supplied command to the device.

6.3.11 SignDeviceCO

Actions: The following actions can be selected from the UI. Scripts cannot use these actions. All these actions take StateMachineContext as an input parameter.

Initialize	Implementation of abstract method. Do nothing.
Clear Screen	Remove everything from display screen
updateLine	Update a specific line with new value. setLineNum, setLineMsg and setLineColor must called from script before this action.
Reset	Send “reset” command to the device.
Post Message Displayed	Post “message displayed” event.

Script Methods

Script methods are those which can be used in the PreScript and PostScript writing while configuring the EdgeProcess.

- init (): associate configured device with this controlled object.
- setEventProvider (): set SignDeviceEP as event provider for this controlled object.
- getEventProvider (): return event provider object.
- setEventProvider (SignDeviceEP eventProvider): set event provider object.
- updateLine (int linenum, String line, String color): send update line command to the device with supplied parameters.
- updateLine (int linenum, String line, String color, String size): send update line command to the device with supplied parameters.

- printTextFile (String textfile, String message, String color, String size): send write text file command to the device with supplied parameters.
- setLineNum (int num): set local variable for current line number.
- setLineMsg (String msg): set local variable for current message.
- setLineColor (String color): set local variable for current color.
- increaseCount (): increase count by 1.
- decreaseCount (): decrease count by 1.
- getCount (): return current count.
- setErrorMessage (String msg): set local variable for error message.
- getErrorMessage (): return current error message.

6.4 Event Providers (EP)

Below are the list of event providers and corresponding events available in those event providers.

These events can be raised depending on the business condition which will be checked in actions in control object and these event provider events will help to transition edge process from one state to another state depending on the type of event raised in the action method.

DeviceEP

- Command Received: provided for edge process.
- Device Connected: provided for edge process.
- Device Disconnected: provided for edge process.
- Device Disabled: sent when device is disabled by controlled object.
- Device Enabled: sent when device is enabled by controlled object.
- Device Read: provided for edge process.
- Device Read Disabled: provided for edge process.
- Device Read Enabled: provided for edge process.

EdgeProcessEP

- Cancel: provided for edge process.
- Cancel Complete: provided for edge process.
- Delay Timer cancelled: sent when delay timer is cancelled.
- Delay timer expired: sent when delay timer is expired.
- Delay timer started: sent when delay timer is started.
- Enter state: provided for edge process.
- Exit state: provided for edge process.
- Process fail: provided for edge process.
- Process in progress: provided for edge process.
- Process Success: provided for edge process.
- Read timer cancelled: sent when read timer is cancelled.

- Read timer expired: sent when read timer is expired.
- Read timer started: sent when read timer is started.
- Reset: provided for edge process.
- Reset completed: sent when reset action completed.
- Reset timer cancelled: sent when reset timer cancelled.
- Reset timer expired: sent when reset timer expired.
- Reset timer started: sent when reset timer started.
- Start read: provided for edge process.
- Stop read: provided for edge process.
- Timer cancelled: sent when custom timer is cancelled.
- Timer expired: sent when custom timer is expired.
- Timer started: sent when custom timer is started.
- To init state: sent when Transition to Init State action is called.
- To next state: sent when Transition to Next State action is called.

External Event

External event can be any string typed in by end user. The application should making sure there is such event be posted to the process zone/edge process so that it can make a proper transition.

PLCDeviceEP

- PLC Read: sent when data is received from PLC DEVICE.
- All eyes off: sent when Clear All Eyes action is called.
- Button pushed: provided for edge process.
- Buzzer off: provided for edge process.
- Buzzer on: provided for edge process.
- Command Received: provided for edge process.
- Device Connected: provided for edge process.
- Device Disabled: sent after Disable action is called.
- Device Disconnected: provided for edge process.
- Device Enabled: sent after Enable action is called.
- Device Read: provided for edge process.
- Device Read Disabled: provided for edge process.
- Device Read Enabled: provided for edge process.
- Entry Sensor off: sent when entry sensor off signal is received.
- Entry sensor on: sent when entry sensor on signal is received.
- Exit sensor off: sent when exit sensor off signal is received.
- Exit sensor on: sent when exit sensor on signal is received.
- Green blink: sent when green blink command is sent.

- Green on: sent when green on command is sent.
- Light off: sent when light off command is sent.
- Orange Blink: sent when orange blink command is sent.
- Orange On: sent when orange on command is sent.
- PLC Read: provided for edge process.
- Red Blink: sent when red blink command is sent.
- Red On: sent when red on command is sent.
- Timer cancelled: sent when custom timer is cancelled.
- Timer expired: sent when custom timer is expired.
- Timer started: sent when custom timer is started.

ReaderDeviceEP

- Command Received: provided for edge process.
- Device Connected: provided for edge process.
- Device Disabled: sent after Disable action is called.
- Device Disconnected: provided for edge process.
- Device Enabled: sent after Enable action is called.
- Device Read: provided for edge process.
- Device Read disabled: provided for edge process.
- Device read Enabled: provided for edge process.
- Tag Read: sent when data received from TAG READER.

ScaleDeviceEP

- Command Received: provided for edge process.
- Device Connected: provided for edge process.
- Device Disabled: sent after Disable action is called.
- Device Disconnected: provided for edge process.
- Device Enabled: sent after Enable action is called.
- Device Read: provided for edge process.
- Device Read disabled: provided for edge process.
- Device read Enabled: provided for edge process.
- Scale read: sent when data is received from SCALE DEVICE

SignDeviceEP

- Command Received: provided for edge process.

- Device Connected: provided for edge process.
- Device Disabled: sent after Disable action is called.
- Device Disconnected: provided for edge process.
- Device Enabled: sent after Enable action is called.
- Device Read: provided for edge process.
- Device Read disabled: provided for edge process.
- Device read Enabled: provided for edge process.
- Message displayed: provided for edge process.

6.5 Example of Building Edge Process

The use case taken is Packing Process.

6.5.1 **Packing Process**: In the supply chain management when shipping of the goods happens, several items are put into one big box and they will be shipped. In this use case we call each individual items as the child items and box in which we put the items called as parent or master pack. Both master pack and child items will have RFID tag. For parent RFID tag and child RFID tag there will be different tag schemes. For example parent RFID tag can start with 305XXX… and child RFID tag can start with 301XXX…

6.5.2 **Pre Requisite**: The association between parent and child items should exist in the InSync IApp application database for a successful packing process.

During the Packing Process we need to do the following validations:

1. Check scanned tag ids against the Tag Schemes (Child tag scheme or Parent Tag scheme)
2. Check scanned tag records are there in Database.
3. Check for Parent tag is there in scanned list of tags.
4. Check for Missing tags for the scanned Parent tag
5. Check for Extra tags for scanned Parent tag
6. Check for scanned tags proper association between Parent Tag and Child tags exists in the database.

If all the above validations are passed, then Packing Process will be successful. In each of these cases we need to display the information (success or failure) to the user so that user can know whether packing process is success or failure.

This is an ideal scenario to solve using the Edge Process. Following are the steps to build the Control object and Event Provider to execute this use case:

6.6 Creating Control Object

1. Creating Control Object Class: To show up in the UI when new control object should following two things:

- Control object class name should end with CO
- Class should extend com.insync.edgeware.pz.sm.co.EdgeProcessCO class so that by default we get many of the predefined actions

Good Example CollectDataCO extends com.insync.pz.sm.co.EdgeProcessCO

Bad Example CollectData extends com.insync.pz.sm.co.EdgeProcessCO

Good Example SinglePackProcessCO extends com.insync.pz.sm.co.EdgeProcessCO

Bad Example SinglePackProcessCO

We can add properties to the CO class so that we can configure them from UI from Properties tab. Here are the rules for properties

It must be public , so that its accessed from UI

It must start with PROP_ (this will tell what fields to be shown in Properties tab)

It must have an annotation EdgeProcessParameter

@EdgeProcessParameter(provider = "CollectDataCO", defaultValue="", mandatory = EdgeProcessConstant.MANDATORY_FALSE, toolTipValue = "Validation Rule", valueMethod = "")

public static final String PROP_REGEX = "Work Order Rule";

where provider is this class, defaultValue is the value defaulted if user doesn't select a value for this property, mandatory is whether its required or not, toolTipValue is the description of this property as shown as tool tip when you hover over the property in the UI, valueMethod is the implementation method to supply the value list for this property so that they are displayed in drop-down in UI

1. After creating the class we need to add the actions required. To create the desired actions we need to write methods which will have one parameter of type com.evelopers.unimod.runtime.context.StateMachineContext. These actions will be showen on the UI to user to select so that on which event which action has to be taken. These methods should have annotation

@EdgeProcessAnnotation (displayName="Create New Tag")

public void createNewTag(StateMachineContext context)

if annotation is not declared then the method will not be shown in the actions panel

1. In this control object we will have the following actions:

- init (StateMachineContext context): This action will be called when user log in to the Packing Screen or clicks on the Start button on the UI to start Packing Process. In this action (method) we will initialize the all the local variables, so that when packing process starts it won't be having any value for the variables which is set in the previous cycle of packing process.
- validateTagRead (StateMachineContext context): This action will be called for each tag read and checks whether read tag is according to either child or parent tag scheme if yes then it adds to scanned tag list.
- validatePackTagRead (StateMachineContext context): This action will be called after collecting the reads for certain amount of time (Read Timer). In this action first we store parent tag id and child tag id in different list by applying the tag validation with tag schemes. After this we need to do the validations listed above against the information stored in the database, depending on validation we need to display the message to the user on the UI. If packing process is fine then we will be updating the status of the items to Packed in database, so that it will be ready for shipping.

6.7 Creating Event Provider

If events provided in base class com.insync.pz.sm.ep.EdgeProcessEP are not sufficient for the use case we can create our own event provider class so that business specific events can be configured in this. Following are the steps to create event provider:

1. Creating Event Provider Class: The class name should end with EP (case sensitive). It should extend the com.insync.iApp.pz.sm.ep.EdgeProcessEP (which provides many events which are frequently used) which again extends com.insync.edgeware.pz.sm.ep.EventProvider which will provide to send the evens to the

State Machine so that edge process will move to targeted state and it can take proper actions (Control object action) in that state.

2. Defining the events:

   a. Each event is declared as public string. If its not public it wont be shown in the Event drop down in the UI.

   b. Each event field must have annotation

      i. @ParameterAnnotation(displayName="Device Read")

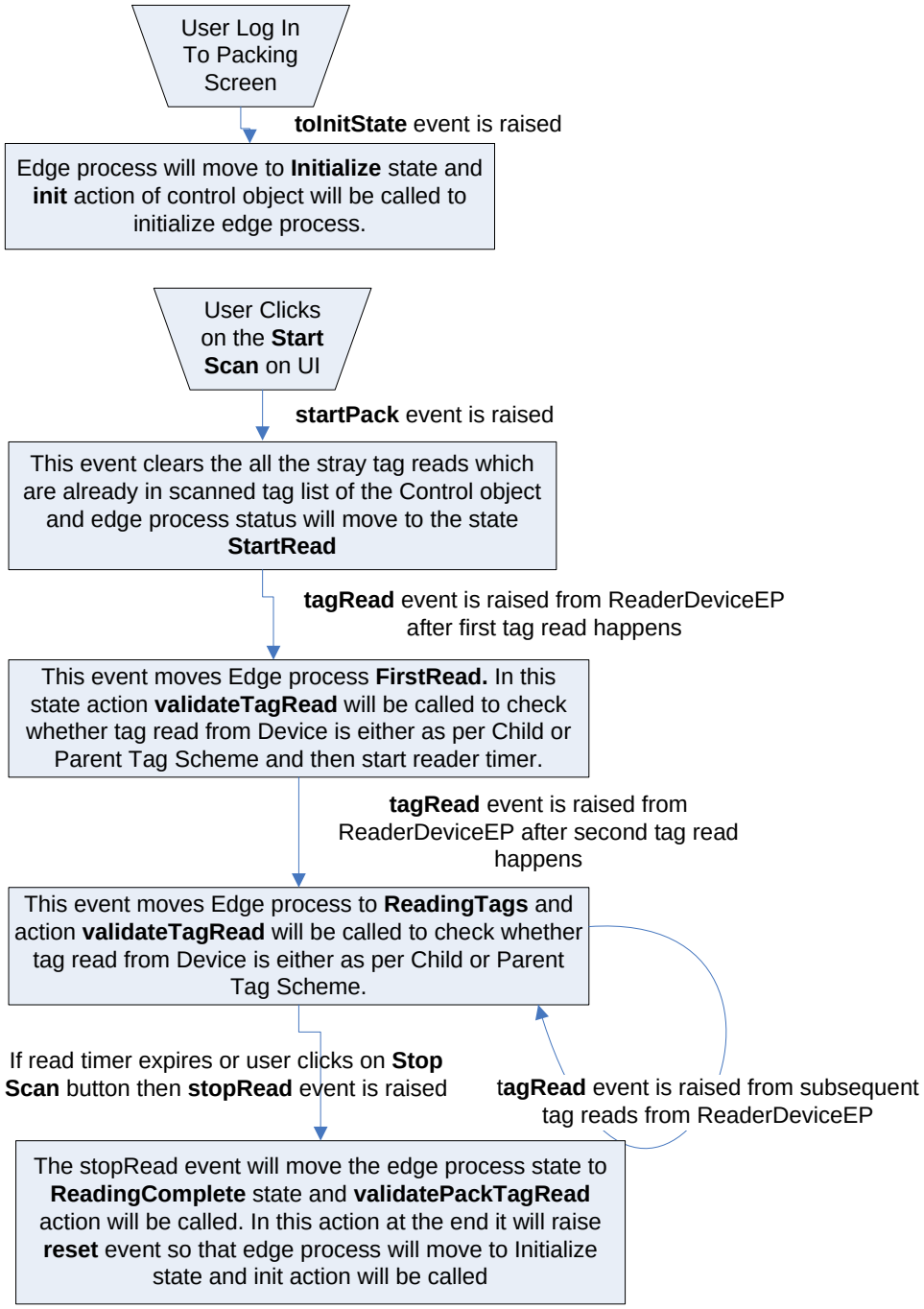
      ii. public static final String DEVICE_READ = "deviceRead";

if annotation is missing, the event will not be shown in the Event drop down in the UI

1. Each event field should have one method which will be calling the super class (com.insync.edgeware.pz.sm.ep.EventProvider) sendEvent (String eventName) so that on edge process can move to targeted next state and take proper actions (Control object action) in that state. These events will be displayed when configuring the edge process on UI. public void deviceRead() {sendEvent(DEVICE_READ);}

Below Diagram shows how the transition of Edge Process (for Packing) happens from one state to another state depending on the events raised and also shows different actions called in particular states:

Packing Process


User Log In To Packing Screen
toInitState event is raised
Edge process will move to Initialize state and init action of control object will be called to initialize edge process.
User Clicks on the Start Scan on UI
startPack event is raised
This event clears the all the stray tag reads which are already in scanned tag list of the Control object and edge process status will move to the state StartRead
tagRead event is raised from ReaderDeviceEP after first tag read happens
This event moves Edge process FirstRead. In this state action validateTagRead will be called to check whether tag read from Device is either as per Child or Parent Tag Scheme and then start reader timer.
tagRead event is raised from ReaderDeviceEP after second tag read happens
This event moves Edge process to ReadingTags and action validateTagRead will be called to check whether tag read from Device is either as per Child or Parent Tag Scheme.
If read timer expires or user clicks on Stop Scan button then stopRead event is raised
tagRead event is raised from subsequent tag reads from ReaderDeviceEP
The stopRead event will move the edge process state to ReadingComplete state and validatePackTagRead action will be called. In this action at the end it will raise reset event so that edge process will move to Initialize state and init action will be called

iii. Section 3
Working with iApp User Interfaces

7 iApp Event Management

7.1 Introduction

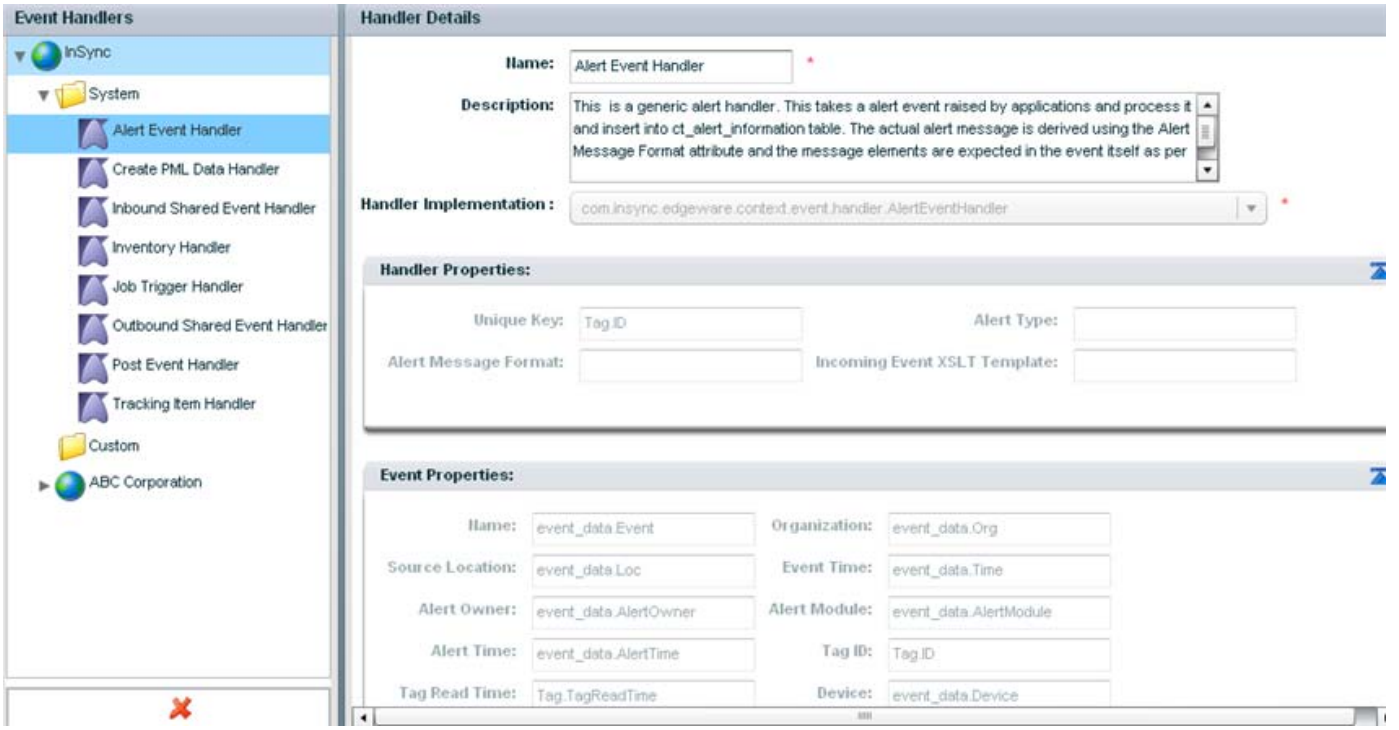
Event Handlers are specific implementation of a task which acts on the Event to take appropriate actions based on the implementation. Event handlers are configured for Events while defining the Actions for the Event.

Each Event handler is a java implementation which understands a set of properties (**Handler Properties**) both internally and as well as on the Event (**Event Properties**) which it will act on if configured for that Event.

Each Handler configured in IApp acts on 2 different sets of data:

- Event Data - Coming from the incoming event.
- Handler Data – Data configured from Sensornet by the user at runtime.

7.2 Handler Properties



Handlers configured in Sensornet have a Handler implementation which is a Java class implementation. The business logic of what each handler should do is built into the implementation with the handler properties and the event properties as the input and exposed to the outside world.

7.3 Event Properties

Event Properties section has the event properties of the incoming event which lets the handler about the structure of the incoming event.

7.4 Handler Properties Section

Handler Properties section has set of properties that user can pass to the handler implementation class at run time. This gives the user the flexibility to override the data coming from event and also make the handler to perform different actions based on the business need.

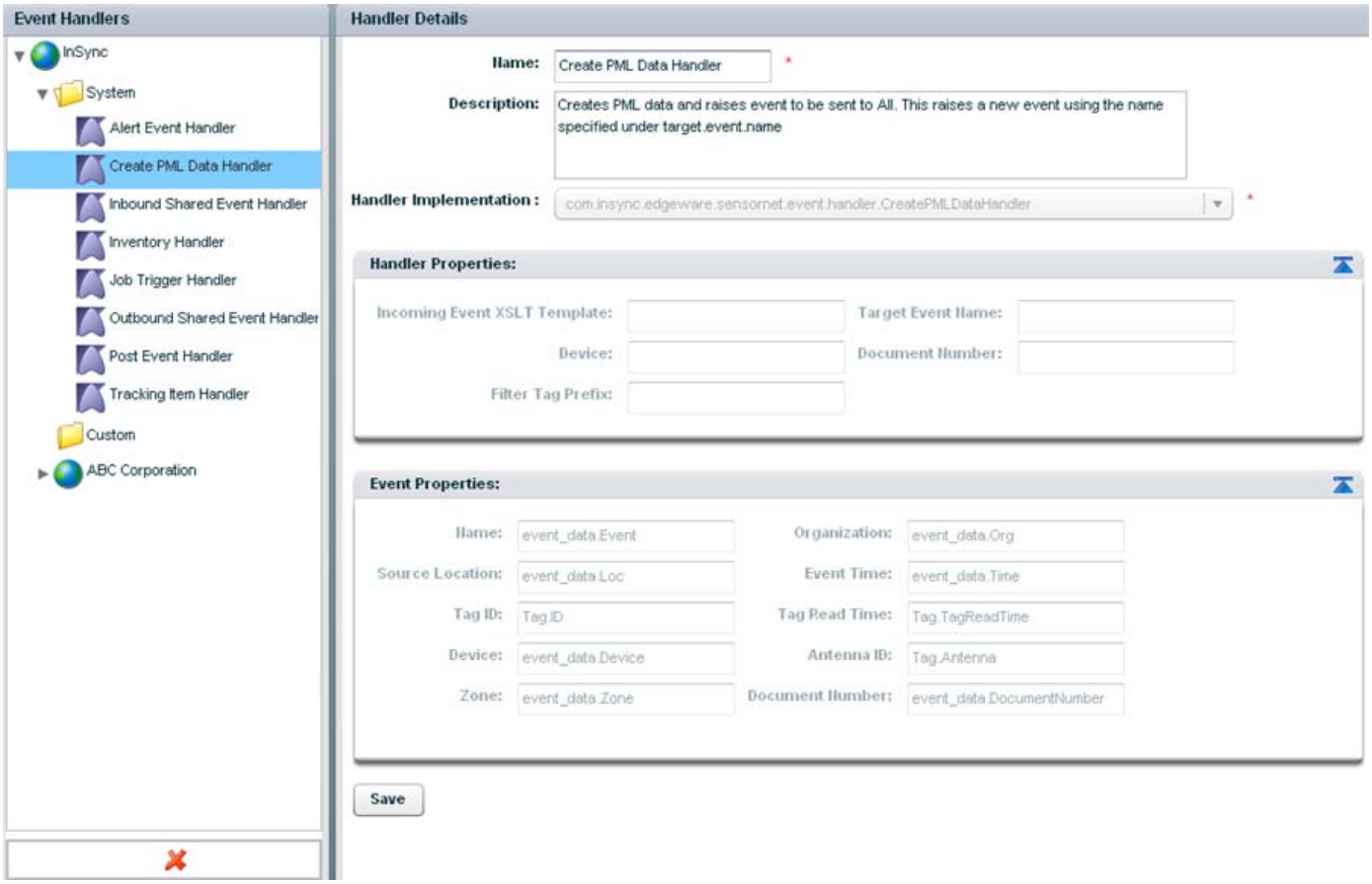
7.5 Event Handlers Available

The Event Handlers available by default in IApp are:

- **Create PML Data Handler:** Creates PML data and raises event to be sent to AII. This raises a new event using the name specified under target.event.name.
- **Alert Event Handler:** This is a generic alert handler. This takes an alert event raised by applications, processes it and inserts into ct_alert_information table. The actual alert message is derived using the Alert Message Format attribute and the message elements are expected in the event itself as per the formatter elements.
- **Inbound Shared Event Handler**: The handler takes the event from trusted partner and processes it internally. If data is coming in from a partner who is not trusted then the event will be ignored.
- **Outbound Shared Event Handler:** Checks the trust settings, creates and posts an event to the trusted partner.
- **Job Trigger Handler:** This will trigger the job to execute after the Trigger Time is expired. When this handler is invoked, it will set the next fire tome to current time plus trigger time.
- **Item Tracking Handler**: This will track the current state of Trackable Item using the Unique Key. Valid values for Unique Key are ID, Serial Number, Tracking Number, but this field must exist in the incoming event.
- **Item History Tracking Handler**: This will track the history of Trackable Item using the Unique Key. Valid values for Unique Key are ID, SerialNumber, TrackingNumber, but this field must exist in the incoming event.
  If Trackable is On Change, then if any of item attribute is updated then only trackable history will be created , if it is Always then trackable
  history is created always, if it is Never then trackable history will not be created.
- **Post Event Handler**: Creates and posts a new event based on certain conditions specified. event.name is used for the new event name. Event. location if supplied overwrites the location name.
- **Inventory Handler**: Updates the inventory of the items based on Action. Valid Actions are RECV, MOVE and SHIP. Inventory is added for RECV, deducted for SHIP and location is changed for MOVE.

7.6 Create PML Data Handler

This handler creates event data in PML format and sends to SAP AII



Handler Properties

Attribute	Comments
Incoming Event XSLT Template	xsl stylesheet used to convert the incoming event data xml so that the handler can understand it
Target Event Name	The final event that will be sent to SAP AII
Device	Filter for device name
Document Number	Document number to be sent to SAP
Filter Tag Prefix	To trim the tag id when used

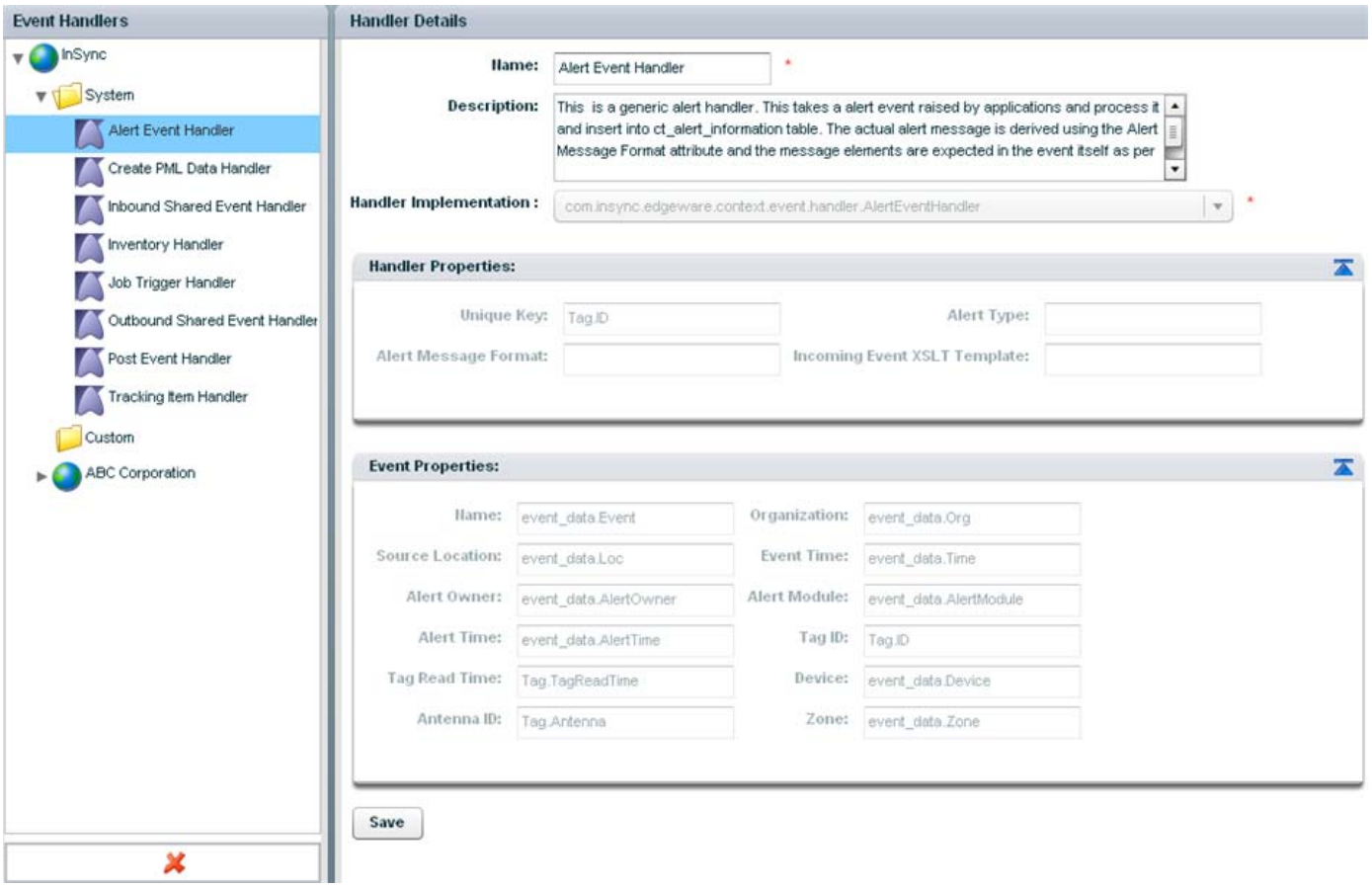
Event Properties

Attribute	Comments
Name	Event Name
Source Location	Location where the event is raised
Event Time	Timestamp when event is raised
Tag Read Time	Read time for a specific tag
Tag ID	Tag ID
Device	RFID Device Name

Antenna ID	Antenna ID where tag is read
Zone	Process Zone name
Document Number	Document number against which this read is happening
Organization	Organization of the location

7.7 Alert Event Handler

This handler takes the event data and persists to al_alert_information table so that the alert information is shown in the reports. It takes any event data and converts the alert message using the xslt formatter.



Handler Properties

Attribute	Comments
Unique key	The unique key attributes from the incoming event data. This key will be used to identify the trackable item.
Alert Type	Type of alert. Informational or actionable. If actionable, action to close the alert is provided in the alert report
Alert Message Format	sl stylesheet to be used to format the alert message.
Incoming Event XSLT Template	The xsl style sheet to use to convert the incoming event data xml so that the handler

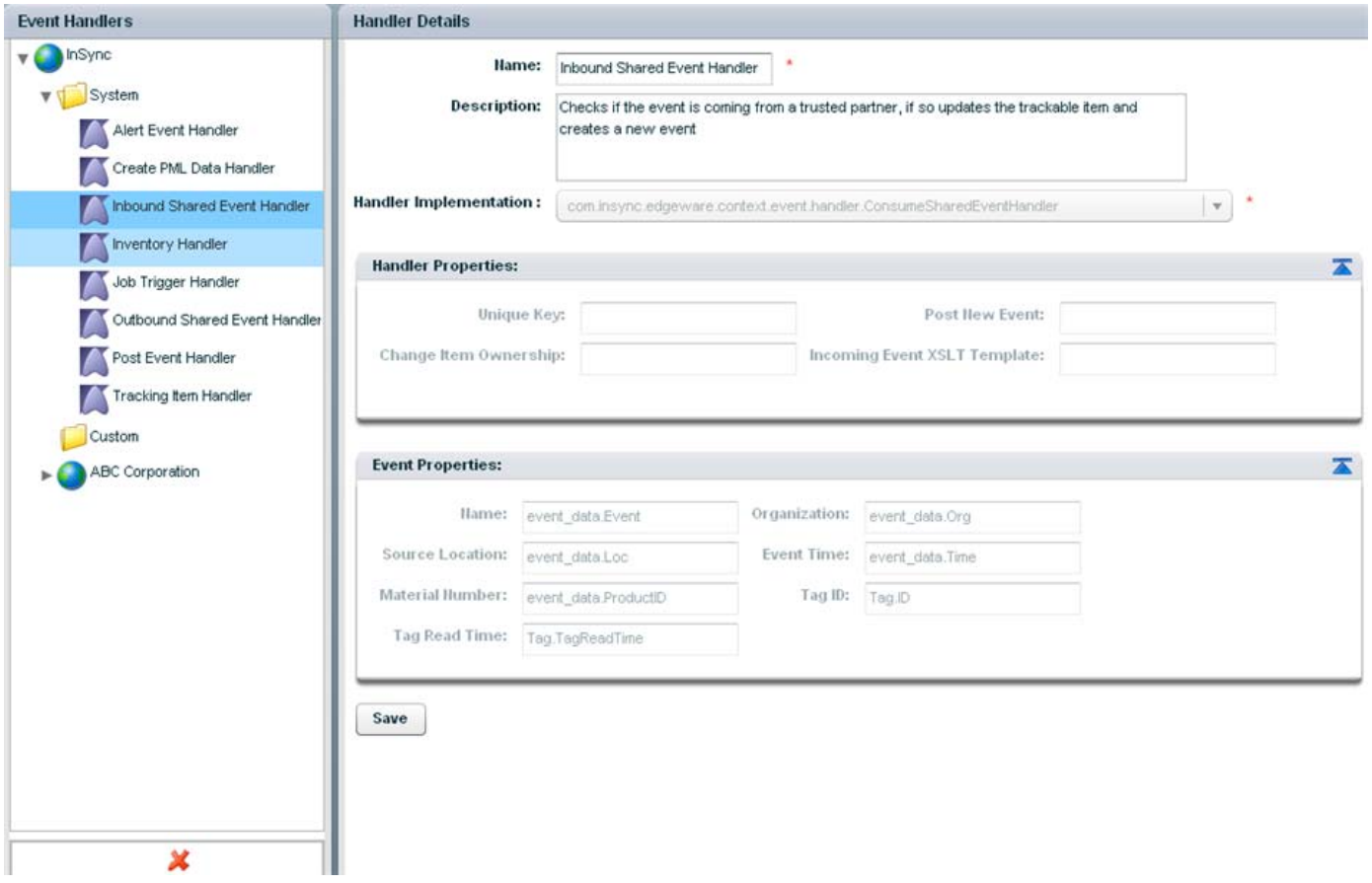
	can understand it

Event Properties

Attribute	Comments
Alert Owner	Owner of the alert who receives it
Alert Module	Module of the system where this alert is raised. Example : Retail Solution
Alert Time	When the alert is raised
Name	Event Name
Source Location	Location where the event is raised
Event Time	Timestamp when event is raised
Tag Read Time	Read time for a specific tag
Tag ID	Tag ID
Device	RFID Device Name
Antenna ID	Antenna ID where tag is read
Zone	Process Zone name
Organization	Organization of the location

7.8 Inbound Shared Event Handler

This handler takes the event from trusted partner and processes it internally. If data is coming from a partner which is not a trusted one then the event will be ignored. It sets the partner data in ct_trackable_item and posts a new event so that it can be processed internally



Handler Properties

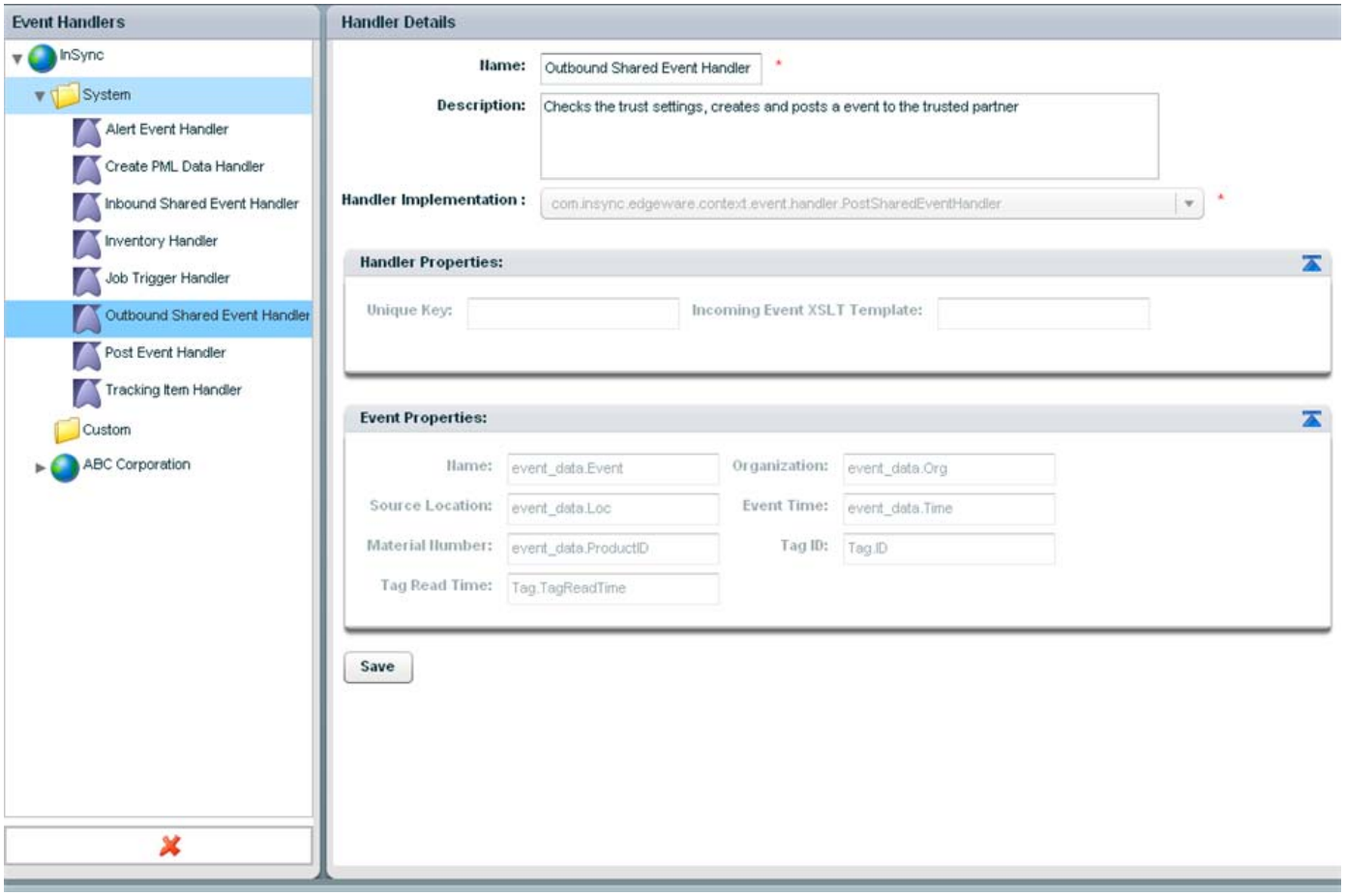
Attribute	Comments
Unique key	The unique key attribute from the incoming event data. This key will be used to identify the trackable item
Post New Event	Event to be posted from this trusted partner
Change item Ownership	If set to true, the item ownership changes from partner to this organization.
Incoming Event XSLT Template	The xslt stylesheet to use to convert the incoming event data xml so that the handler can understand it

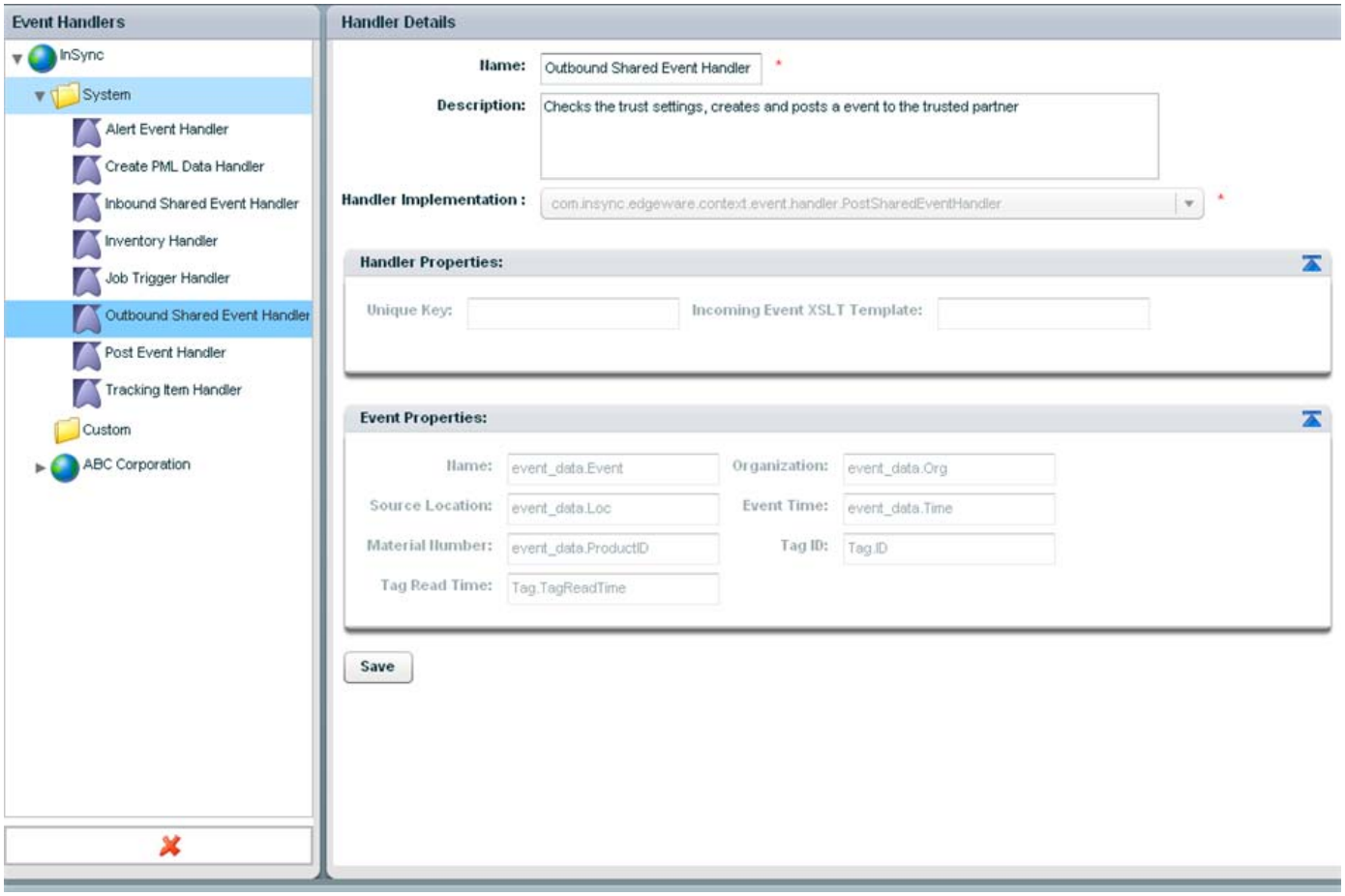
Event Properties

Attribute	Comments
Name	Event Name
Source Location	Location where the event is raised
Event Time	Timestamp when event is raised
Tag ID	Tag ID
Material Number	Material number of the product
Tag Read Time	Read time for a specific tag
Organization	Organization of the location

7.9 Outbound Shared Event Handler

This handler posts event to trusted partner. The event name to be posted is derived from shared events map. Before raising event, it checks if the partner is trusted or not and the event to be shared or not.


Event Handlers
InSync
System
Alert Event Handler
Create PML Data Handler
Inbound Shared Event Handler
Inventory Handler
Job Trigger Handler
Outbound Shared Event Handler
Post Event Handler
Tracking Item Handler
Custom
ABC Corporation
Handler Details
Name:
Outbound Shared Event Handler
Description:
Checks the trust settings, creates and posts a event to the trusted partner
Handler Implementation :
com.insync.edgeware.context.event.handler.PostSharedEventHandler
Handler Properties:
Unique Key:
Incoming Event XSLT Template:
Event Properties:
Name:
event_data.Event
Organization:
event_data.Org
Source Location:
event_data.Loc
Event Time:
event_data.Time
Material Number:
event_data.ProductID
Tag ID:
Tag.ID
Tag Read Time:
Tag.TagReadTime
Save



Handler Properties

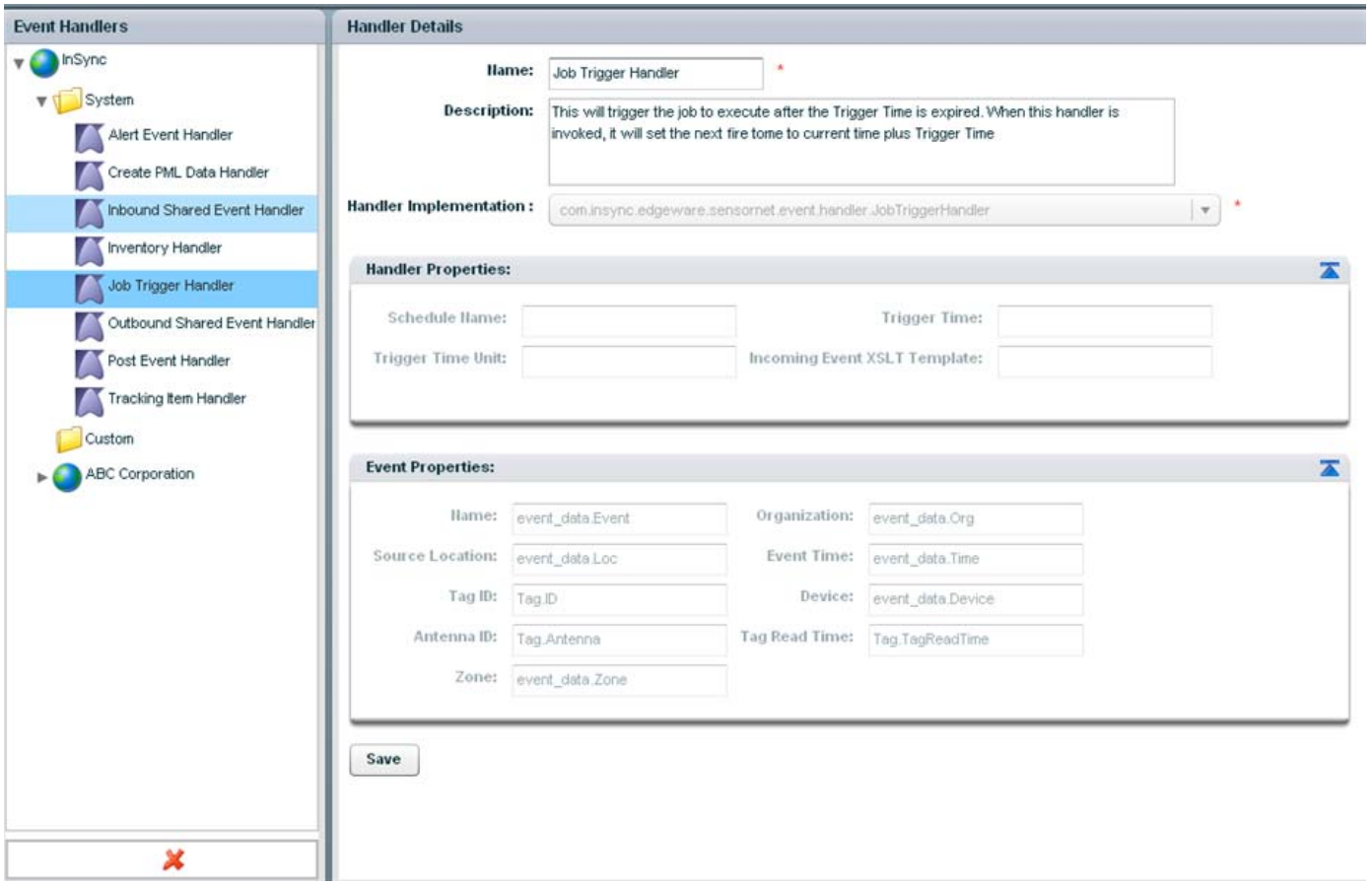
Attribute	Comments
Unique key	The unique key attributes from the incoming event data. This key will be used to identify the trackable item
Incoming Event XSLT Template	The xslt style sheet to use to convert the incoming event data xml so that the handler can understand it

Event Properties

Attribute	Comments
Name	Event Name
Source Location	Location where the event is raised
Event Time	Timestamp when event is raised
Tag ID	Tag ID
Material Number	Material number of the product
Tag Read Time	Read time for a specific tag
Organization	Organization of the location

7.10 Job Trigger Handler

This handler will trigger the job, even if the job is scheduled at a later date. After this trigger, the job will run as per original schedule. This handler makes sure the job is invoked whenever we want



Handler Properties

Attribute	Comments
Schedule Name	Name of the schedule which needs to be triggered
Trigger Time	Time at which the trigger must fire AFTER this handler is invoked.
Trigger Time Unit	Time unit for Trigger Time
Incoming Event XSLT Template	The xsl stylesheet to use to convert the incoming event data xml so that the handler can understand it

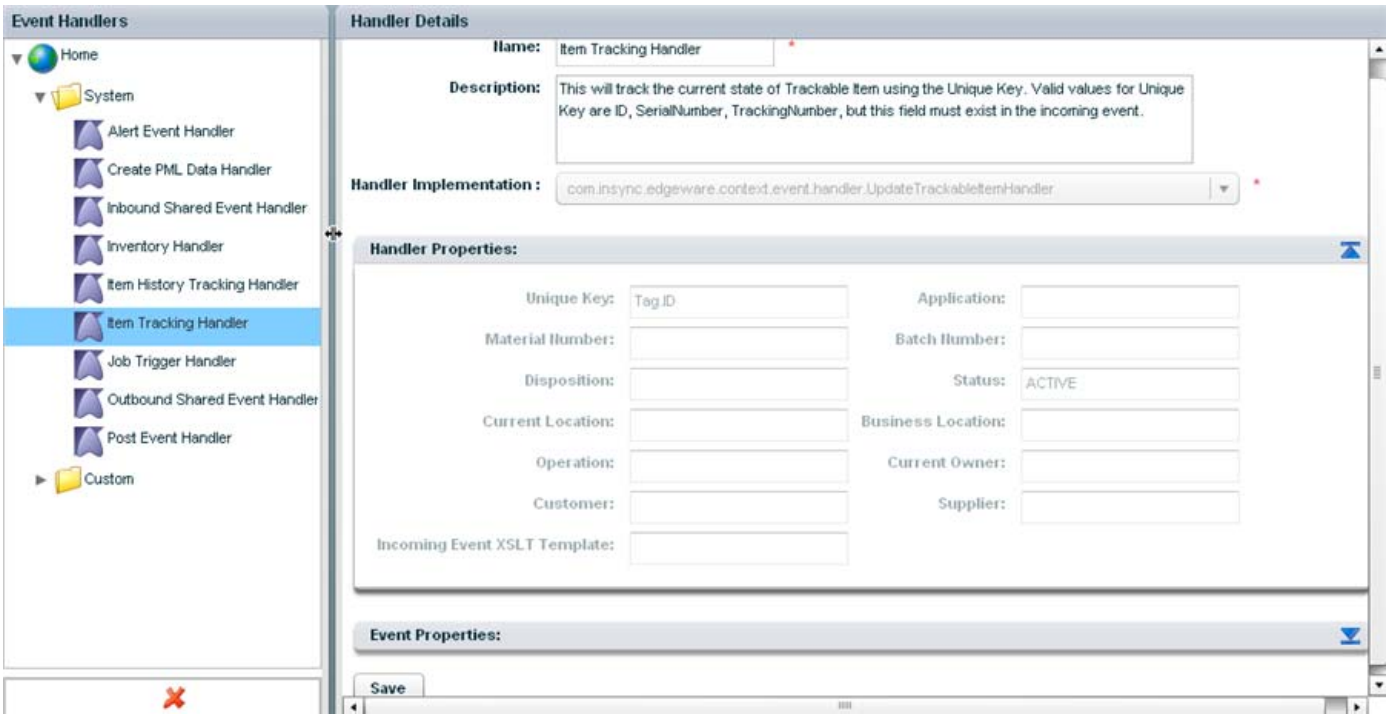
Event Properties

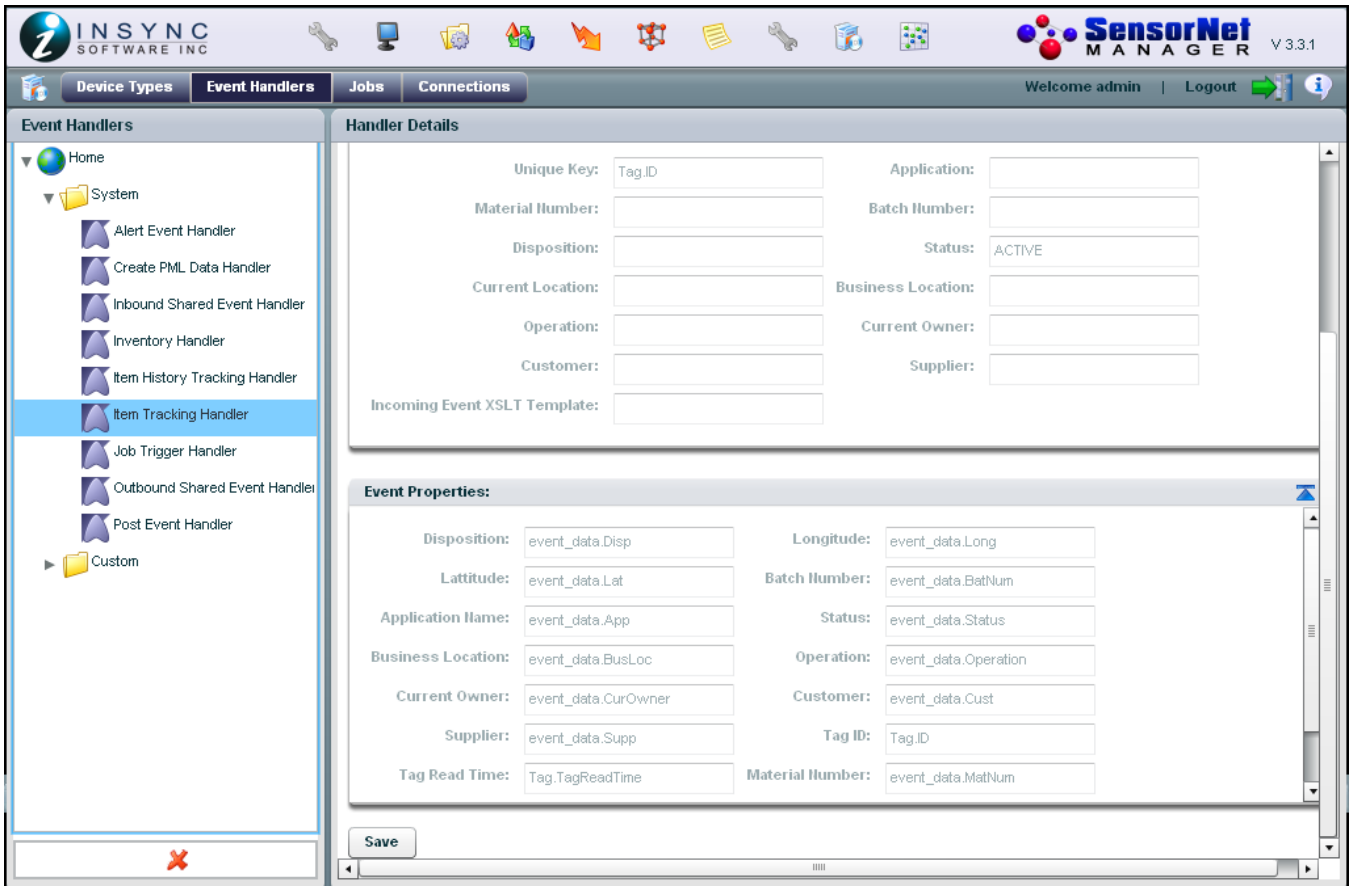
Attribute	Comments
Name	Event Name
Source Location	Location where the event is raised
Event Time	Timestamp when event is raised
Tag ID	Tag ID

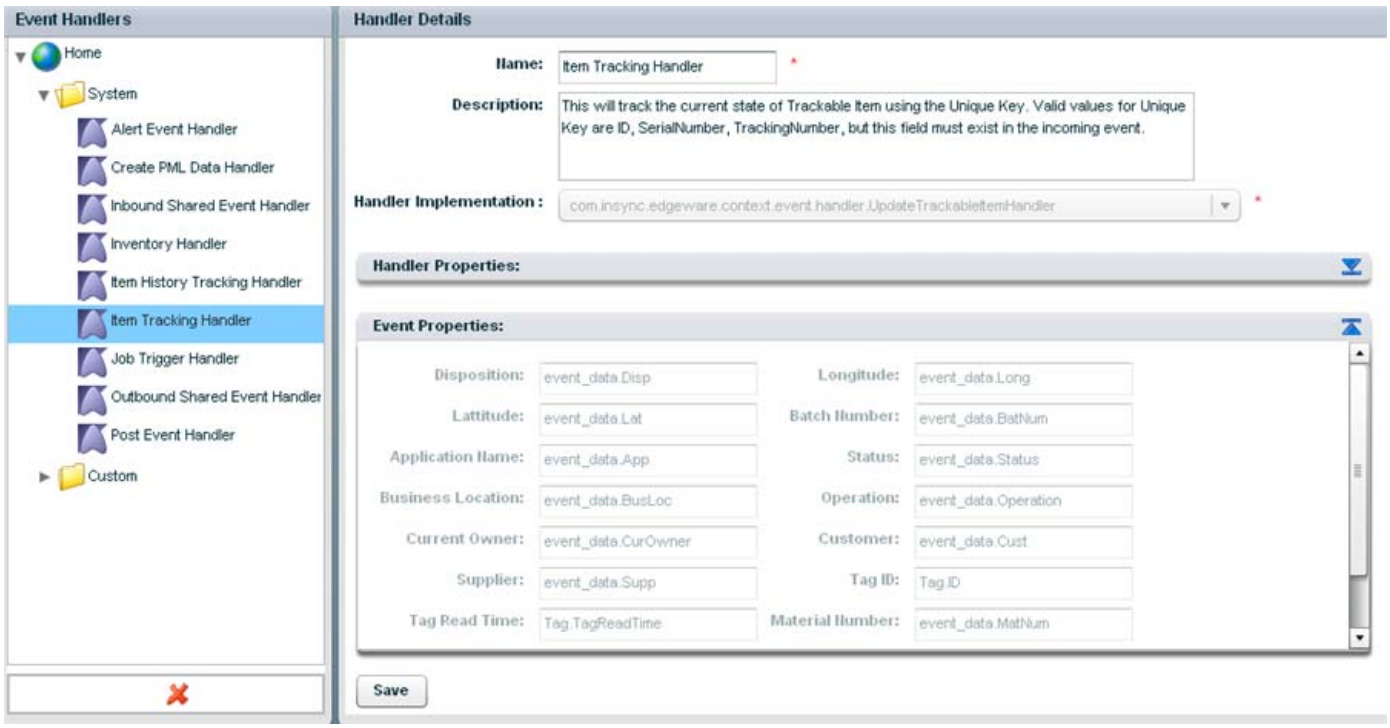
Antenna ID	Antenna ID where tag is read
Zone	Process Zone name
Organization	Organization of the location
Tag Read Time	Read time for a specific tag
Device	RFID Reader name

7.11 Item Tracking Handler

This handler will create/update trackable item in the database. If we want to track an item, we need to first create this. If item already exists then it will be updated







Handler Properties

Attribute	Comments

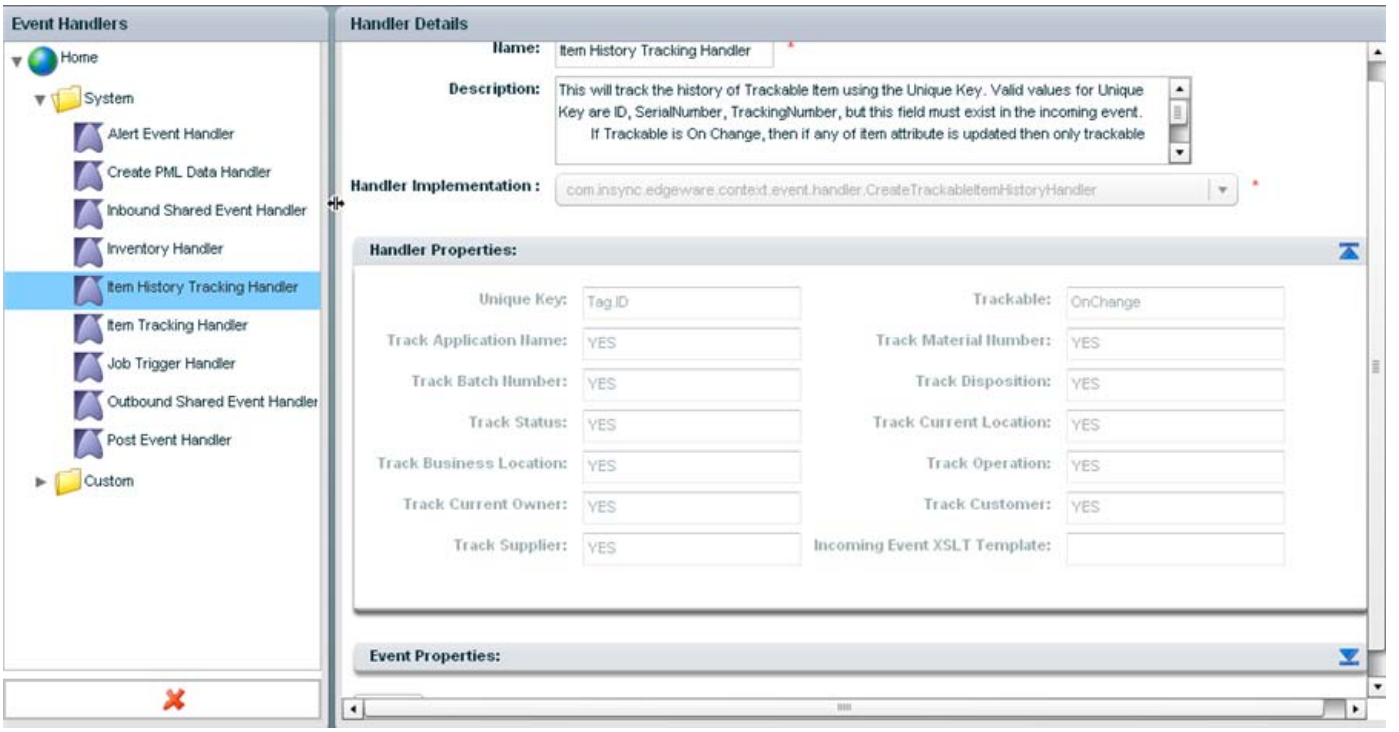
Unique key	The unique key attribute from the incoming event data. This key will be used to identify the trackable item
Incoming Event XSLT Template	xsl stylesheet to use to convert the incoming event data xml so that the handler can understand it
Application	Input the Application name to be added to the event while processing the event handler
Material Number	Input the Material Number to be added to the event while processing the event handler
Batch Number	Input the Batch Number to be added to the event while processing the event handler
Disposition	Input the Disposition to be added to the event while processing the event handler
Status	Input the Status to be added to the event while processing the event handler
Current Location	Input the Current Location to be added to the event while processing the event handler
Business Location	Input the Business Location to be added to the event while processing the event handler
Operation	Input the Operation to be added to the event while processing the event handler
Current Owner	Input the Current Owner to be added to the event while processing the event handler
Customer	Input the Customer to be added to the event while processing the event handler
Supplier	Input the Supplier to be added to the event while processing the event handler

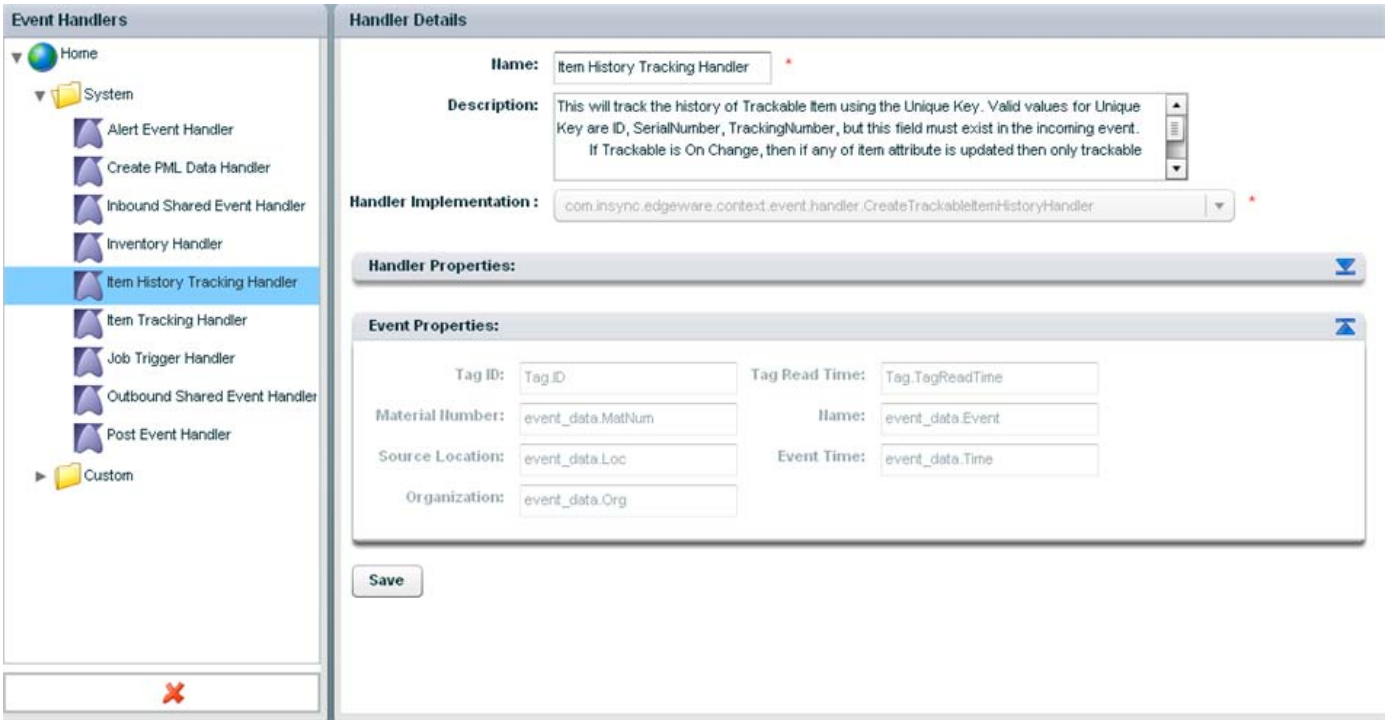
Event Properties

Attribute	Comments
Disposition	Antenna ID where tag is read
Latitude	Location co-ordinate for latitude
Longitude	Location co-ordinate for longitude
Batch Number	Batch Number of the item
Application Name	Application for which event is raised
Status	Status of the item
Business Location	Business Location for which event is raised
Operation	Operation for which event is raised
Current Owner	Current owner of the item
Customer	Customer of the item
Supplier	Supplier of the item
Tag ID	Tag ID
Tag Read Time	Read time for a specific tag
Material Number	Material number of the product
Name	Event Name
Source Location	Location where the event is raised
Event Time	Timestamp when event is raised
Organization	Organization of the location

7.12 Item History Tracking Handler

This will track the history of Trackable Item using the Unique Key. Valid values for Unique Key are ID, SerialNumber, TrackingNumber, but this field must exist in the incoming event.
If Trackable is On Change, then if any of item attribute is updated then only trackable history will be created. If it is Always then trackable history is created always, if it is Never then trackable history will not be created


Event Handlers
Home
System
Alert Event Handler
Create PML Data Handler
Inbound Shared Event Handler
Inventory Handler
Item History Tracking Handler
Item Tracking Handler
Job Trigger Handler
Outbound Shared Event Handler
Post Event Handler
Custom
Handler Details
Name: Item History Tracking Handler
Description: This will track the history of Trackable Item using the Unique Key. Valid values for Unique Key are ID, SerialNumber, TrackingNumber, but this field must exist in the incoming event. If Trackable is On Change, then if any of item attribute is updated then only trackable
Handler Implementation : com.insync.edgeware.context.event.handler.CreateTrackableItemHistoryHandler
Handler Properties:
Unique Key: Tag.ID
Trackable: OnChange
Track Application Name: YES
Track Material Number: YES
Track Batch Number: YES
Track Disposition: YES
Track Status: YES
Track Current Location: YES
Track Business Location: YES
Track Operation: YES
Track Current Owner: YES
Track Customer: YES
Track Supplier: YES
Incoming Event XSLT Template:
Event Properties:


Event Handlers
Home
System
Alert Event Handler
Create PML Data Handler
Inbound Shared Event Handler
Inventory Handler
Item History Tracking Handler
Item Tracking Handler
Job Trigger Handler
Outbound Shared Event Handler
Post Event Handler
Custom
Handler Details
Name: Item History Tracking Handler
Description: This will track the history of Trackable Item using the Unique Key. Valid values for Unique Key are ID, SerialNumber, TrackingNumber, but this field must exist in the incoming event. If Trackable is On Change, then if any of item attribute is updated then only trackable
Handler Implementation : com.insync.edgeware.context.event.handler.CreateTrackableItemHistoryHandler
Handler Properties:
Event Properties:
Tag ID: Tag.ID
Tag Read Time: Tag.TagReadTime
Material Number: event_data.MatNum
Name: event_data.Event
Source Location: event_data.Loc
Event Time: event_data.Time
Organization: event_data.Org
Save

Handler Properties

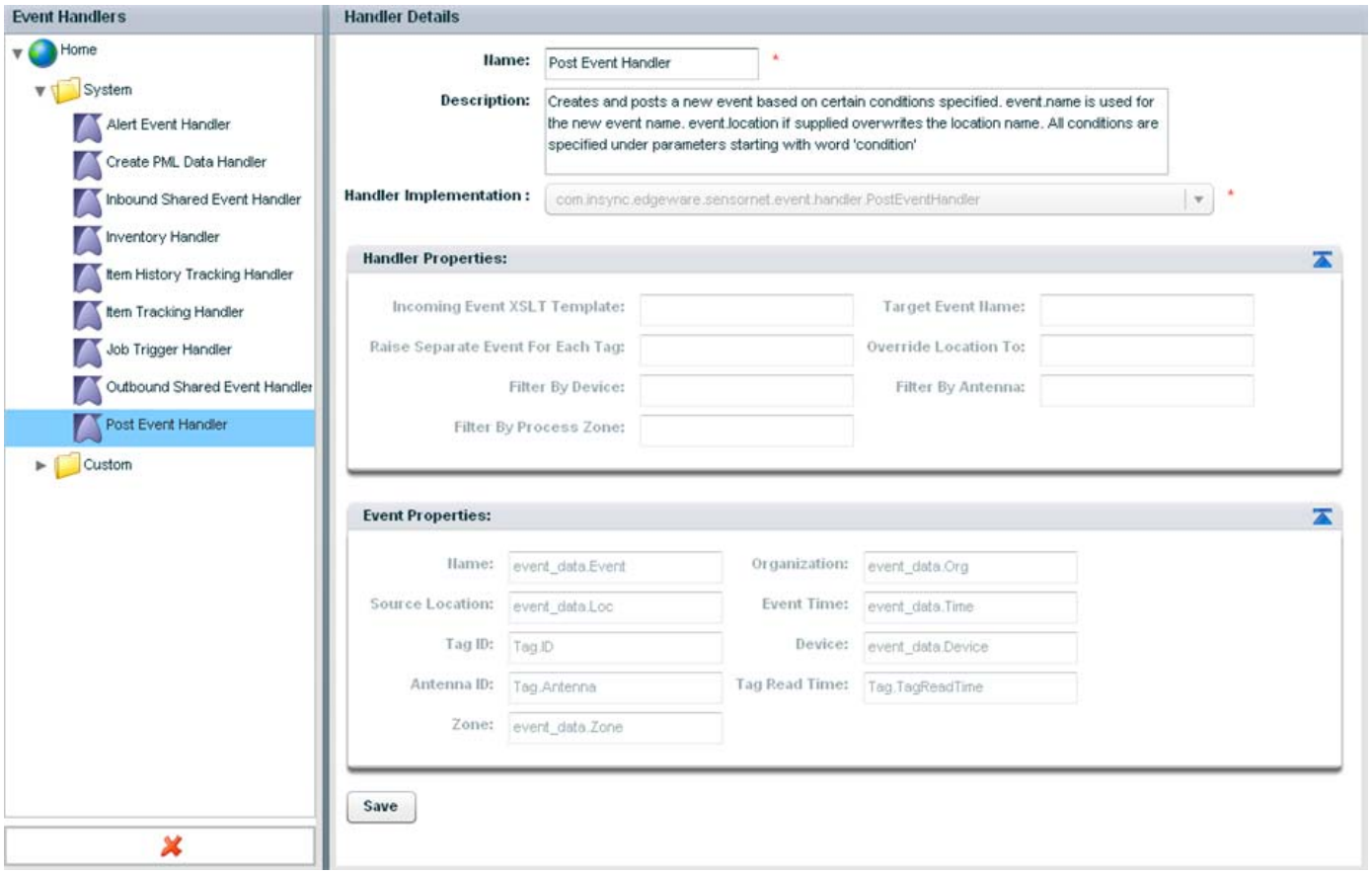
Attribute	Comments
Unique key	The unique key attribute from the incoming event data. This key will be used to identify the trackable item
Incoming Event XSLT Template	The xslt stylesheet to use to convert the incoming event data xml so that the handler can understand it
Trackable	This attribute determines whether item history will be tracked 'On Change' or 'Always'.
Track Application Name	This attribute determines whether Item History should be created on change of Application Name.
Track Material Number	This attribute determines whether Item History should be created on change of Material Number.
Track Batch Number	This attribute determines whether Item History should be created on change of Batch Number.
Track Disposition	This attribute determines whether Item History should be created on change of Disposition.
Track Status	This attribute determines whether Item History should be created on change of Status.
Track Current Location	This attribute determines whether Item History should be created on change of Current Location.
Track Business Location	This attribute determines whether Item History should be created on change of Business Location.
Track Operation	This attribute determines whether Item History should be created on change of Operation.
Track Current Owner	This attribute determines whether Item History should be created on change of Current Owner.
Track Customer	This attribute determines whether Item History should be created on change of Customer.
Track Supplier	This attribute determines whether Item History should be created on change of Supplier.

Event Properties

Attribute	Comments
Tag ID	Tag ID
Tag Read Time	Read time for a specific tag
Material Number	Material number of the product
Name	Event Name
Source Location	Location where the event is raised
Event Time	Timestamp when event is raised
Organization	Organization of the location

7.13 Post Event Handler

This handler will post a new event from the incoming event data. This will be useful to redirect the event with a new name, without changing the data structure.



Handler Properties

Attribute	Comments
Incoming Event XSLT Template	The xslt stylesheet to use to convert the incoming event data xml so that the handler can understand it
Target Event Name	The event to be posted
Override Location To	If you want to override the location name in the new event
Raise Separate Event For Each Tag	In case of multiple tags , raise Separate event for each tag or same event for all tags.
Filter By Device	If used new event will be posted only when the device is matching with the filter
Filter By Antenna	If used new event will be posted only when the antenna is matching with the filter
Filter By Process Zone	If used new event will be posted only when the Process Zone is matching with the filter

Event Properties

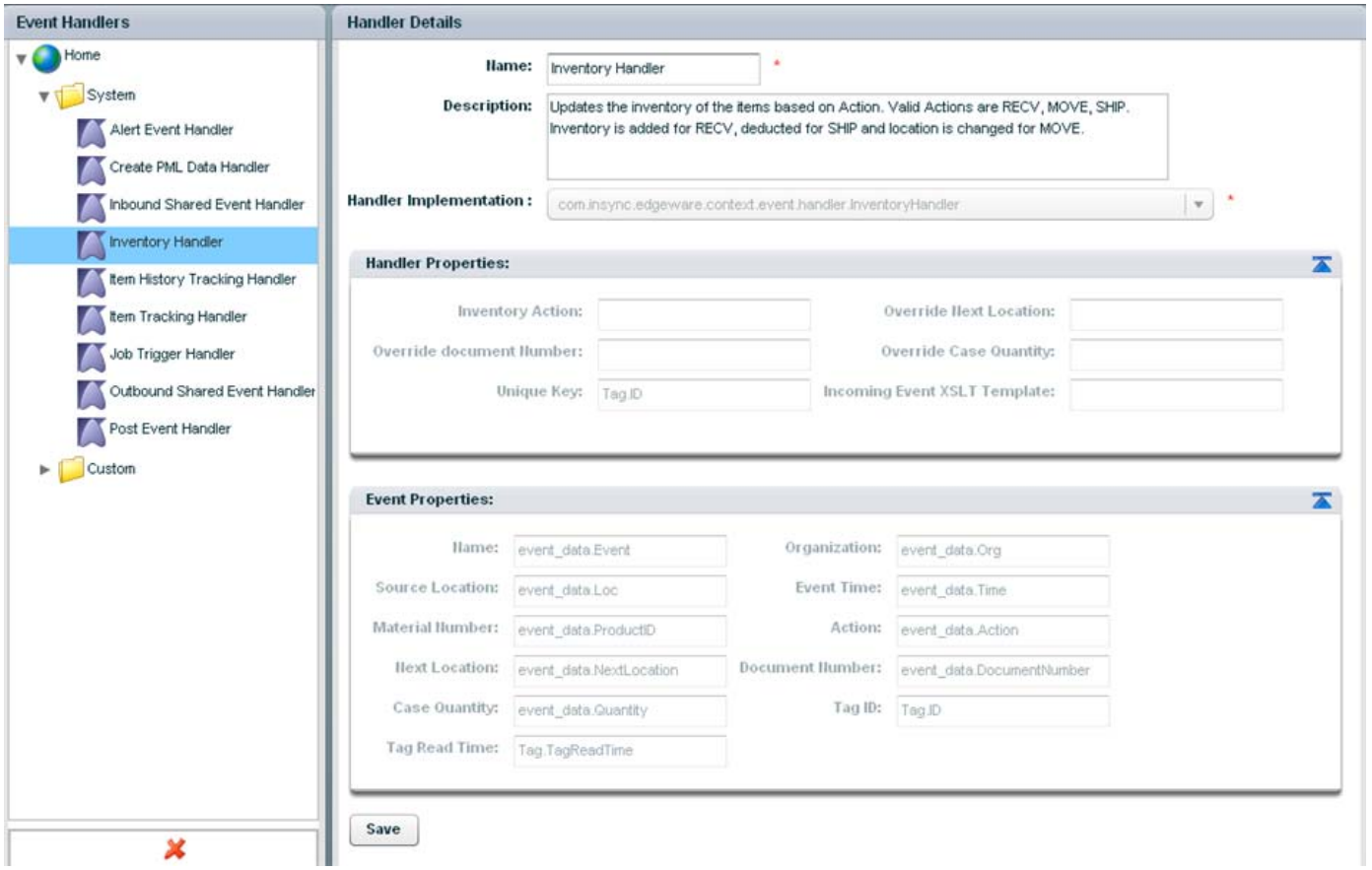
Attribute	Comments
Name	Event Name
Source Location	Location where the event is raised
Event Time	Timestamp when event is raised
Tag ID	Tag ID

Antenna ID	Antenna ID where tag is read
Zone	Process Zone name
Organization	Organization of the location
Tag Read Time	Read time for a specific tag
Device	RFID Reader name

7.14 Inventory Handler

The Inventory Handler is required for inventory management. The Handler updates the item inventory at the location where the item is received or moved or shipped.
It is required to be configured at locations/read points where the inventory needs to be managed.

Note: It is required that the Create Item Handler is configured for Inventory Handler.



Handler Properties

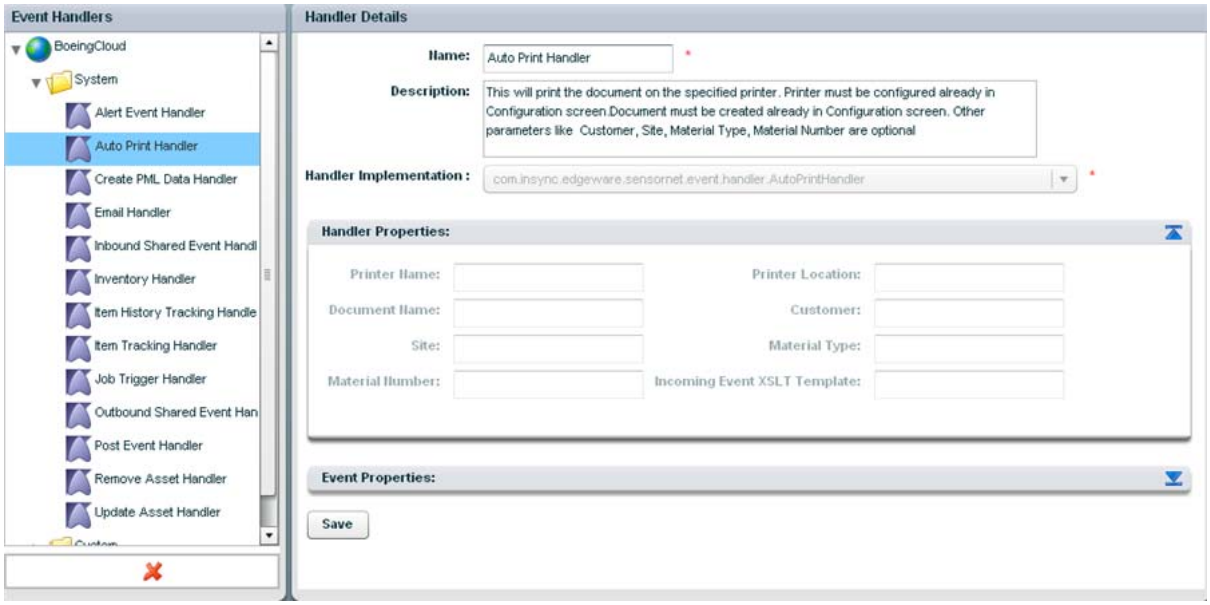
Attribute	Comments
Inventory Action	Action to be taken : RECV, MOVE, SHIP
Override Next Location	In case of ship, we can give the next location
Override Document Number	Document number to used in case of SHIP
Override Case Quantity	Quantity in the case
Unique Key	Key to identify the item
Incoming Event XSLT Template	The xslt style sheet to use to convert the incoming event data xml so that the handler can understand it

Event Properties

Attribute	Comments
Name	Event Name
Source Location	Location where the event is raised
Event Time	Timestamp when event is raised
Tag ID	Tag ID
Next Location	In case of ship, next location
Case Quantity	Quantity in case
Organization	Organization of the location
Document Number	Document number to used in case of SHIP
Tag Read Time	Read time for a specific tag
Action	Inventory action like RECV, MOV, SHIP
Product Type	Product ID for this item

7.15 Auto Print Handler

The Auto Print Handler creates a Print Event which goes through the document creation process based on the attributes set on the event by the handler and prints the document on the specified printer.



Handler Properties

Attribute	Comments
Printer Name	Lists the printer drivers registered on the server where the print server is hosted
Printer Location	Lists the locations created in the location configuration
Site	Site name at which the printer is located
Material Type	Material type of the product
Material Number	Material number of the product

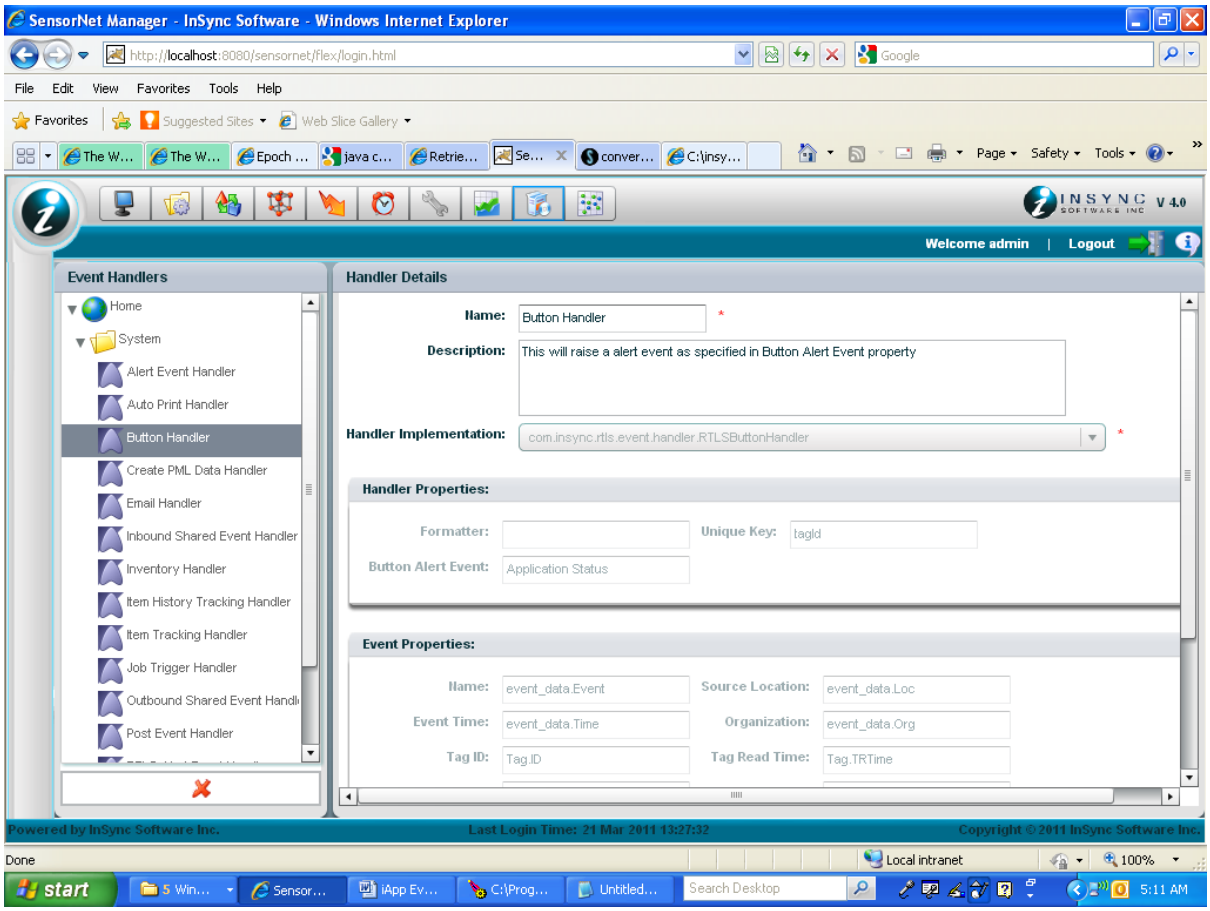
Incoming Event XSLT Template	The xslt style sheet to use to convert the incoming event data xml so that the handler can understand it

Event Properties

Attribute	Comments
Printer Name	Name of the printer
Printer Location	Location of the printer
Document Name	Name of the printer document
Customer	Name of the customer
Site	Site at which the printer is located
Material Type	Material type of the product
Material Number	Material number of the product
Tag ID	Tag ID of the product
Device	RFID Reader name
Antenna ID	Antenna ID where tag is read
Tag Read Time	Read time for a specific tag
Zone	Process Zone name
Name	Name of the event
Source Location	Location where the event is raised
Event Time	Timestamp when event is raised
Organization	Organization of the location

7.16 Button Handler

The Button Handler raises an alert event as specified in Button Alert Event property.



Handler Properties

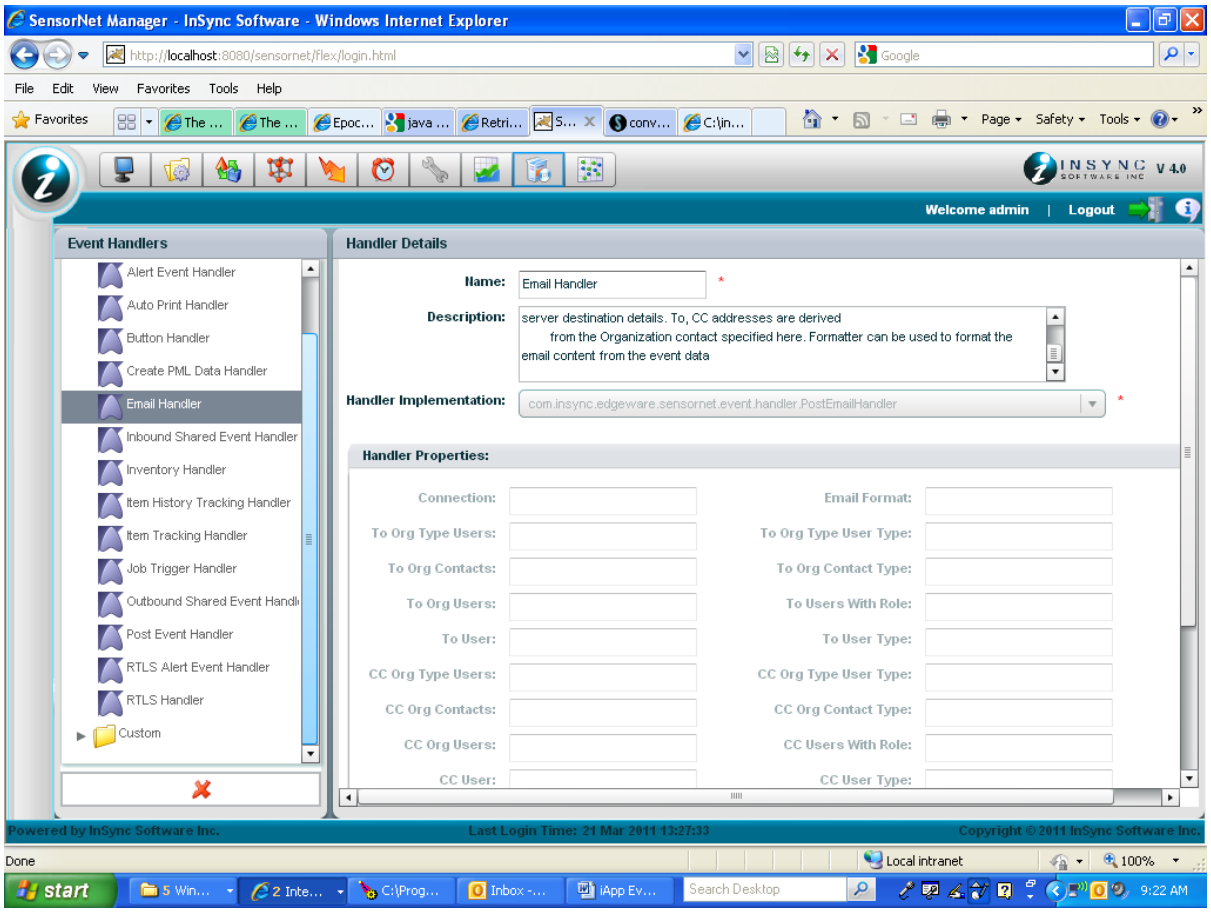
Attribute	Comments
Formatter	The formatter that needs to be used.
Unique Key	The unique key attributes from the incoming event data. This key will be used to identify the trackable item.
Button Alert Event	The Event that needs to be raised by this handler.

Event Properties

Attribute	Comments
Name	EventName
Source Location	Location where the event is raised
Event Time	Timestamp when event is raised
Organization	Organization of the location
Tag ID	Tag ID of the product
Tag Read Time	Read time for a specific tag
Material Number	Material number of the product
Unique Field	Unique Field that is used to identify the tag.
Button Alert Event	Event that needs to be raised.
Button	Button Name.
Action	Action to be taken when button is pressed.

7.17 Email Handler

Email Handlers sends email based on the data received from the event, handler and the connection object. This is achieved by creating a connection of type Email in Administration, Connections screen. Use that connection name here to get the mail server destination details. To, CC addresses are derived from the Organization contact specified here. Formatter can be used to format the email content from the event data.


SensorNet Manager - InSync Software - Windows Internet Explorer
Welcome admin | Logout
Event Handlers
Alert Event Handler
Auto Print Handler
Button Handler
Create PML Data Handler
Email Handler
Inbound Shared Event Handler
Inventory Handler
Item History Tracking Handler
Item Tracking Handler
Job Trigger Handler
Post Event Handler
RTLS Alert Event Handler
RTLS Handler
Custom
Handler Details
Name: Email Handler
Description: server destination details. To, CC addresses are derived from the Organization contact specified here. Formatter can be used to format the email content from the event data
Handler Implementation: com.insync.edgeware.sensornet.event.handler.PostEmailHandler
Handler Properties:
Connection:
Email Format:
To Org Type Users:
To Org Type User Type:
To Org Contacts:
To Org Contact Type:
To Org Users:
To Users With Role:
To User:
To User Type:
CC Org Type Users:
CC Org Type User Type:
CC Org Contacts:
CC Org Contact Type:
CC Org Users:
CC Users With Role:
CC User:
CC User Type:
Powered by InSync Software Inc.
Last Login Time: 21 Mar 2011 13:27:33
Copyright © 2011 InSync Software Inc.

Handler Properties

Attribute	Comments
Connection	Email Connection containing the smtp server and other details to send email.
Email Format	Format that needs to be used to format the email content.
To Org Type Users	
To Org Type User Format	
To Org Contacts	This lists the organization types for this enterprise in the drop down, We can send email to organization contact of this org
To Org Contact Type	From the contacts selected from 'To Org Contacts' further filtering can be done by contact type.
To Org Users	This lists the organization types for this enterprise in the drop down, We can send email to all users in this org.

To Users with Role	This lists all roles in that enterprise, We can send email to all users belonging to this role.
To User	This lists all users in this enterprise; We can send email to this user.
To User Type	This lists the user types in this enterprise, email will be sent to users of this type.
CC Org Type Users	
CC Org Type User Type	
CC Org Contacts	This lists the organization types for this enterprise in the drop down, We can send copy of email to organization contact of this org.
CC Org Contact Type	From the contacts selected from 'CC Org Contacts' further filtering can be done by contact type. A Copy of the email will be sent only to those selected Contact Type.
CC Org Users	This lists the organization types for this enterprise in the drop down, We can send copy of email to all users in this org.
CC Users with Role	This lists all roles in that enterprise, We can send copy of this email to all users belonging to this role.
CC User	This lists all users in this enterprise; We can send copy of this email to this user.
CC User Type	This lists the user types in this enterprise, copy of email will be sent to users of this type.
Subject	Email Subject.
Incoming Event XSLT Template	The XSLT template that needs to be applied to the incoming event can be specified.

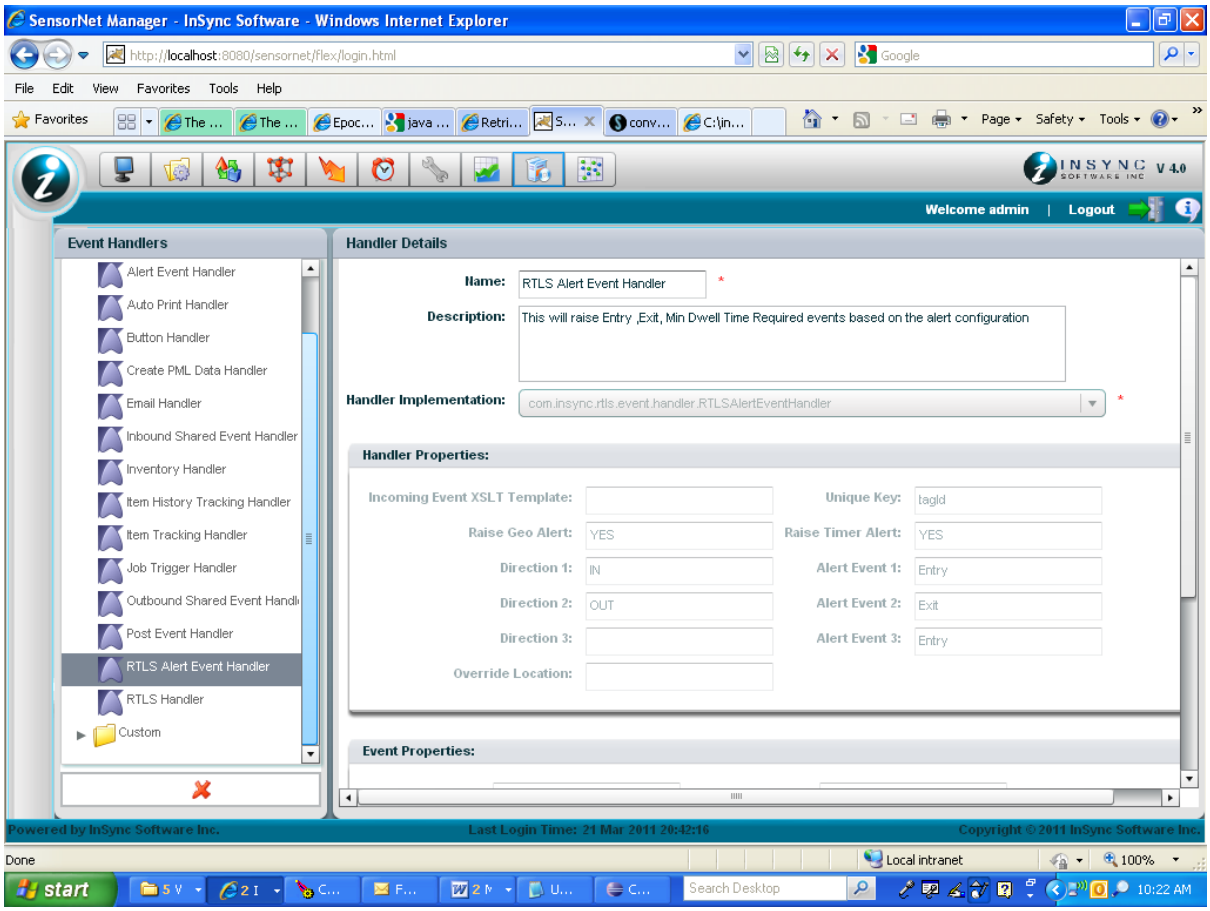
Event Properties

Attribute	Comments
Connections	Email Connection containing the smtp server and other details to send email.
Email Format	Format that needs to be used to format the email content.
To Org Type Users	To all users of the organization of this type in this enterprise.
To Org User Type	To all users of this type in this enterprise.
To Org Contacts	This lists the organization types for this enterprise in the drop down, We can send email to organization contact of this org
To Org Contact Type	From the contacts selected from 'To Org Contacts' further filtering can be done by contact type.
To Org Users	This lists the organization types for this enterprise in the drop down, We can send email to all users in this org.
To Users with Role	This lists all roles in that enterprise, We can send email to all users belonging to this role
To User	This lists all users in this enterprise; We can send email to this user.
To User Type	This lists the user types in this enterprise, email will be sent to users of this type.
CC Org Type Users	A Copy of the email will be sent to all users of

	the organization of this type in this enterprise.
CC Org User Type	A Copy of the email will be sent to all users of this type in this enterprise.
CC Org Contacts	This lists the organization types for this enterprise in the drop down, We can send copy of email to organization contact of this org.
CC Org Contact Type	From the contacts selected from 'CC Org Contacts' further filtering can be done by contact type. A Copy of the email will be sent only to those selected Contact Type.
CC Org Users	This lists the organization types for this enterprise in the drop down, We can send copy of email to all users in this org.
CC Users with Role	This lists all roles in that enterprise, We can send copy of this email to all users belonging to this role.
CC User	This lists all users in this enterprise; We can send copy of this email to this user.
CC User Type	This lists the user types in this enterprise, copy of email will be sent to users of this type.
Subject	Email Subject
Tag ID	Tag ID of the product
Device	RFID Device Name
Antenna Id	Antenna ID where tag is read
Tag Read Time	Read time for a specific tag
Zone	Process Zone name
Name	Event Name
Source Location	Location where the event is raised
Event Time	Timestamp when event is raised
Organization	Organization of the location

7.18 RTLS Alert Event Handler

This will raise Entry, Exit, Min Dwell Time Required events based on the alert configuration.



Handler Properties

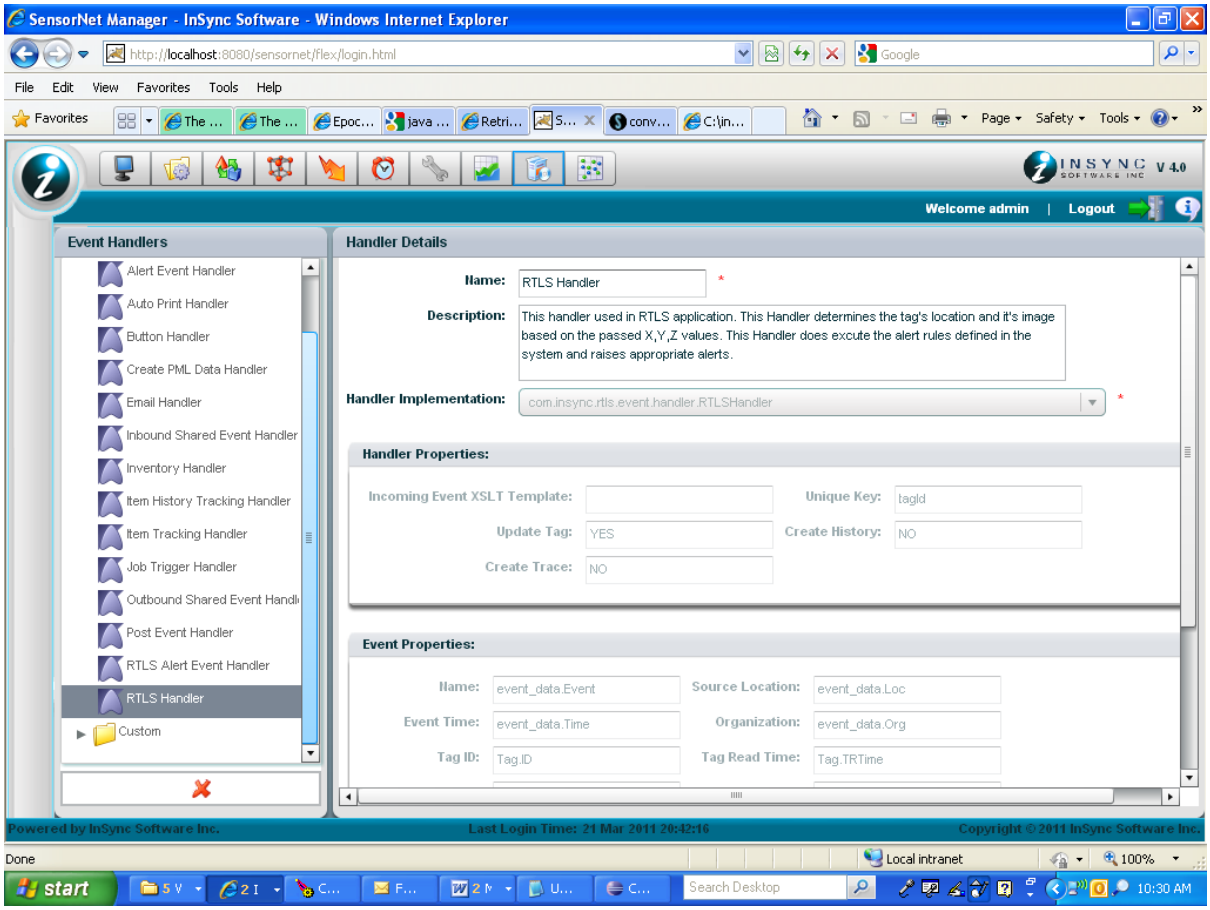
Attribute	Comments
Incoming Event XSLT Template	The XSLT template that needs to be applied to the incoming event can be specified.
Unique Key	The unique key attributes from the incoming event data. This key will be used to identify the trackable item.
Raise Geo Alert	Whether Geo Alert Should be raised or not.
Raise Timer Alert	Whether Timer Alert Should be raised or not.
Direction 1	Direction to be used. Whether IN or OUT
Direction 2	Direction to be used. Whether IN or OUT
Direction 3	Direction to be used. Whether IN or OUT
Alert Event 1	The type of alert event to raise. Example Entry or Exit alert.
Alert Event 2	The type of alert event to raise. Example Entry or Exit alert.
Alert Event 3	The type of alert event to raise. Example Entry or Exit alert.
Override Location	Location to override the incoming location value from the event.

Event Properties

Attribute	Comments
Name	EventName
Source Location	Location where the event is raised
Event Time	Timestamp when event is raised
Organization	Organization of the location
Tag ID	Tag ID of the product
Tag Read Time	Read time for a specific tag
Material Number	Material number of the product
Unique Field	Unique Field that is used to identify the tag.
Direction	Direction to be used.
Alert Event Name	Alert Event that will be raised.

7.19 RTLS Handler

This handler used in RTLS application. This Handler determines the tag's location and it's image based on the passed X, Y, Z values. This Handler does execute the alert rules defined in the system and raises appropriate alerts.


SensorNet Manager - InSync Software - Windows Internet Explorer
Welcome admin | Logout
Event Handlers
Alert Event Handler
Auto Print Handler
Button Handler
Create PML Data Handler
Email Handler
Inbound Shared Event Handler
Inventory Handler
Item History Tracking Handler
Item Tracking Handler
Job Trigger Handler
Post Event Handler
RTLS Alert Event Handler
RTLS Handler
Custom
Handler Details
Name: RTLS Handler
Description: This handler used in RTLS application. This Handler determines the tag's location and it's image based on the passed X,Y,Z values. This Handler does excute the alert rules defined in the system and raises appropriate alerts.
Handler Implementation: com.insync.rtls.event.handler.RTLSHandler
Handler Properties:
Incoming Event XSLT Template:
Unique Key: tagId
Update Tag: YES
Create History: NO
Create Trace: NO
Event Properties:
Name: event_data.Event
Source Location: event_data.Loc
Event Time: event_data.Time
Organization: event_data.Org
Tag ID: Tag.ID
Tag Read Time: Tag.TRTime
Powered by InSync Software Inc.
Last Login Time: 21 Mar 2011 20:42:16
Copyright © 2011 InSync Software Inc.

Handler Properties

Attribute	Comments
Incoming Event XSLT Template	The XSLT template that needs to be applied to the incoming event can be specified.
Unique Key	The unique key attributes from the incoming

	event data. This key will be used to identify the trackable item.
Update Tag	Tag will be updated based on this attribute.
Create History	Inserts record in history table depending on this parameter value
Create Trace	Creates a trace depending upon this parameter value.

Event Properties

Attribute	Comments
Name	EventName
Source Location	Location where the event is raised
Event Time	Timestamp when event is raised
Organization	Organization of the location
Tag ID	Tag ID of the product
Tag Read Time	Read time for a specific tag
Material Number	Material number of the product
Unique Field	Unique Field that is used to identify the tag.
X Co-ordinate	Tag X Longitude
Y Co-ordinate	Tag Y Latitude
Z Co-ordinate	Tag Z Altitude
Accuracy	Tag Accuracy

7.20 Introduction to Event Handler Design

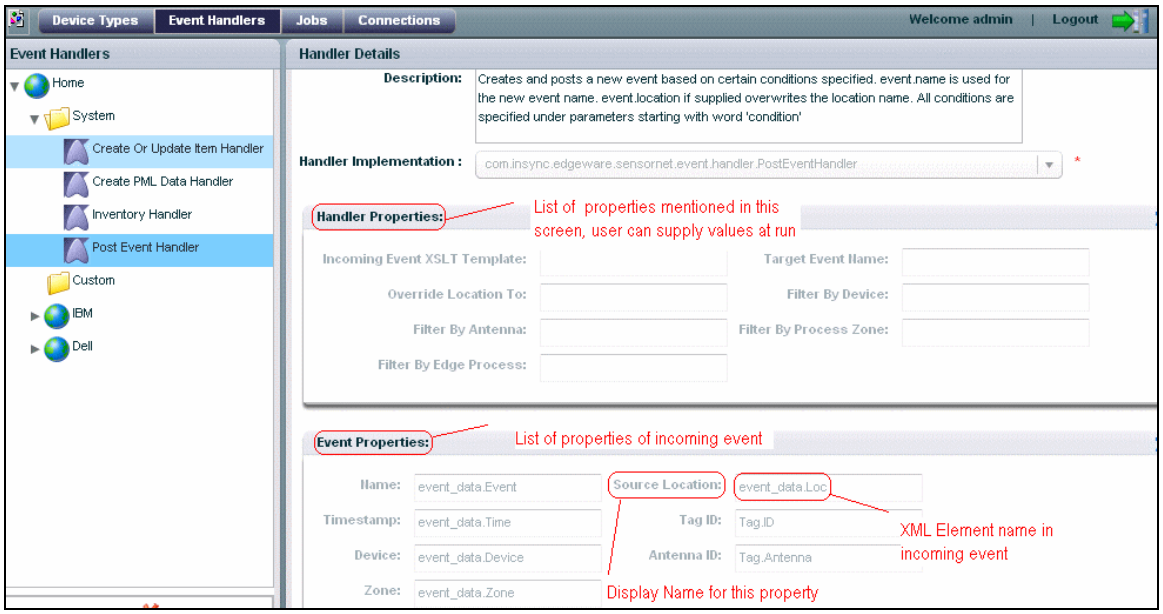
This document describes the structure of the event handlers and the design behind creating a new event handler in iApp. It also shows how to create a new event handler.

7.21 Naming Convention

The Device Types available in iApp are

There are 2 types of data coming to the event handler – one set of data comes from the incoming event and another set of data from the screen (this gives user to modify the values at run time). We call them ‘Event Data’ and ‘Handler Data’. The variables used to identify them in the handler implementation class follow a certain naming convention. If this convention is not followed then the APIs will NOT work. Event Data variables start with **event** and Handler Data variables start with **handler**.

7.22 Event Handler



Taking the example of the Post Event Handler, it has the following 3 sections.

- '**Handler Implementation'** is a java class that implements this handler. This class name should end with 'Handler' and must implement com.insync.ems.service. BaseEventHandler. Examples : UpdatePurchaseOrderHandler, ProcessPackDataHandler, ReceiveItemHandler.

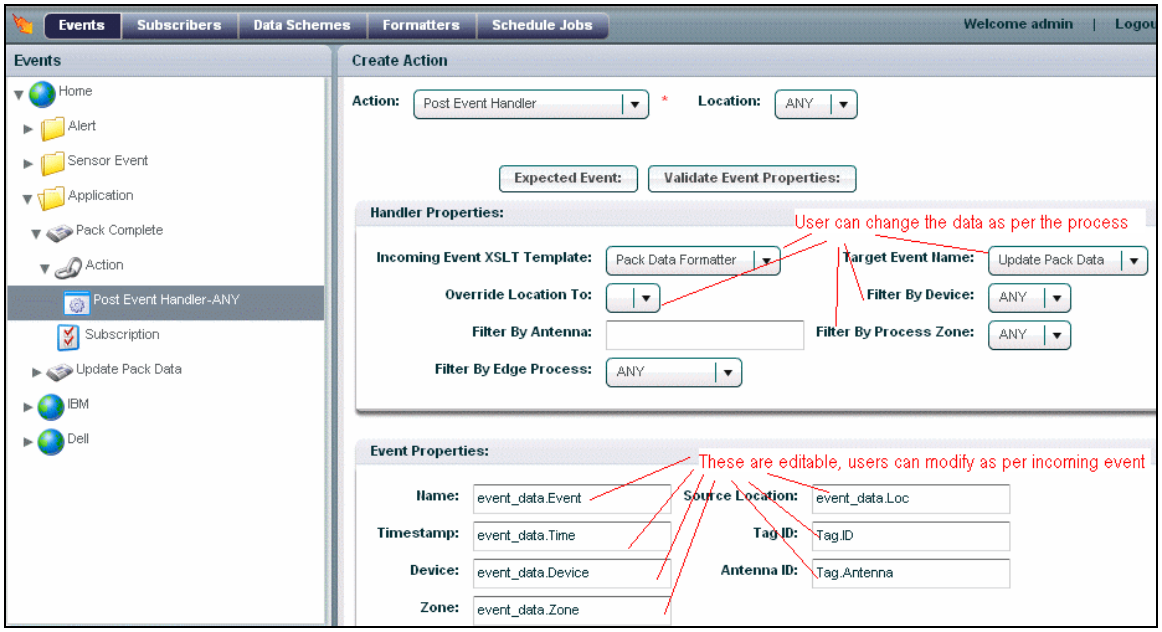
- **Handler Properties** section has set of properties that user can pass to the handler implementation class at run time. This gives the user the flexibility to override the data coming from event and also make the handler to perform different actions based on the business need without changing the code. For example, when you are receiving the items for first time, you can set the inventory action , business step to RECEIVE, item disposition to NEW and when the item is transferred from one stocking location to another then you can set the inventory action, business step to MOVE and item disposition to readyforuse and when the item is shipped out of the warehouse it can be marked as SHIP and transit. In the above handler we have a handler property called 'Incoming Event XSLT Template', this is used to transform the incoming event to the format that the handler can understand. This gives the handler the flexibility to receive any event and transform it to the way it understands. For example in a CreatePOHandler we can receive the PO event from several sources. We can not expect the PO Event to come in a certain format always. Users/Systems can generate PO Event as per their own process. With this XSLT feature we can transform it and use it without changing the code. We just need to create a XSL file to transform the incoming event to the handler format.

- **Event Properties** section has the event properties of incoming event. For example 'Source Location' property corresponds to event_data.Loc, the expected incoming event for this handler should look like

```
<event_data>
        <Loc>Pack Line 1</Loc>
        <Time>12/3/2008 9:34:40 AM</Time>
        <Tag>
                <ID>3034024220649B0000000001</ID>
                <Antenna>1</Antenna>
        </Tag>
        <Device>R1001</Device>
```

```
                <Zone>Pack Zone</Zone>
    </event_data>
```

7.23 Event Action



After the Event Handler is created, user can create actions on this handler. Here user can override/change the handler, event properties as per the process at run time. We don't need to restart the server when values are modified here. This screen has 4 sections.

- **Expected Event**. This shows the structure of the incoming event. If the event comes in a different structure (with XSLT Template) , then the handler implementation will fail.

- **Validate Event Properties**. This is used to validate the event properties as mentioned in 'Event Properties' section. For example the event 'Pack Complete' has a datascheme (dtd/xsd) and the Event Properties doesn't match this datascheme , then the screen will show validation errors – it shows missing properties and additional properties. For example, this event has a datascheme with Source Location defined as event_data.Location and this screen has Source Location as event_data.Loc, then validation fails as the property values are not matching.

- **Handler Properties.** Typically these properties are not editable in Event Handler screen because same handler can be used in different scenarios under different event actions. That's why they are editable here in this screen, user can modify them to get different behavior at runtime without having to change the code.

- **Event Properties**. Typically these properties are not editable in Event Handler screen because same handler can be used to receive the event in different structures. That's why they are editable here in this screen, user can modify them to match with incoming event at runtime without having to change the code.

7.24 APIs

In the handler implementation class, we use java annotation for the property variables. This gives us the flexibility to define multiple dimensions for same variable.

```
public @interface EventActionParameter {
        String defaultValue();
        String valueMethod();
        String displayName();
        String exampleValue();
}
```

```
        @EventActionParameter(defaultValue = "",
displayName=Constants.DISP_HANDLER_EVENT_LOCATION,valueMethod="com.insync.dataobjec
t.service.OrganizationService.getLocationNamesByOrgKey",exampleValue="")
                public static final String HANDLER_EVENT_LOCATION = "handler.override.location";
```

defaultValue → the screen will be filled with this value if supplied
valueMethod → the method to return the data to be filled in the drop-down lists. In the above example, com.insync.dataobject.service.OrganizationService.getLocationNamesByOrgKey returns list of location names to be filled in this property drop-down list
displayName → used as the property label in the screen. Example : Source Location, Incoming Event XSLT Template, Override Location To …
exampleValue → values used in the 'Expected Event' window results

The property variables should follow the naming convention as mentioned above, event properties must have name starting with EVENT_ and the value starting with event. Similarly handler properties must have name starting with HANDLER_ and the value starting with handler.

Event Property

```
@EventActionParameter(defaultValue = Constants.XML_DEVICE_ELEMENT,
displayName=Constants.DISP_EVENT_DEVICE_NAME,valueMethod="",exampleValue="R1
001")
public static final String EVENT_DEVICE_NAME ="event.device;
```

Handler Property

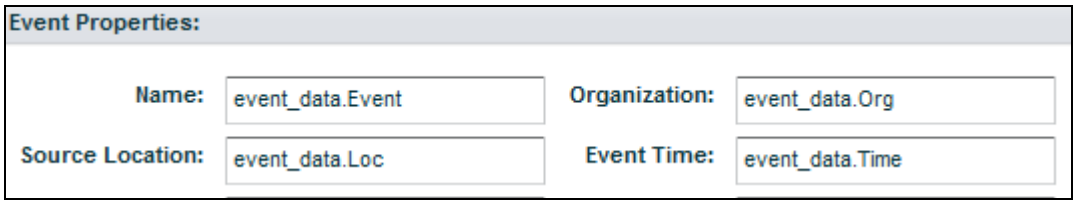
```
@EventActionParameter(defaultValue = "",
displayName=Constants.DISP_HANDLER_EVENT_LOCATION,valueMethod="com.insync.d
ataobject.service.OrganizationService.getLocationNamesByOrgKey",exampleValue="")
public static final String HANDLER_EVENT_LOCATION = "handler.override.location";
```

To read the values of these properties at runtime, you can use this

```
String deviceName = (String) context.getEventActionDataByName(EVENT_DEVICE_NAME);
        String newLocation =  (String)
context.getEventActionDataByName(HANDLER_EVENT_LOCATION);
```

You can use this code to read both event and handler properties

All incoming events MUST have 3 properties Event Name, Source Location, Timestamp. This is required to identify the event and its source/origin and when the event occurred at the source/origin



To get these values in the handler implementation class, you can call

getEventName()
getEventLocation()
getEventDate()

These are implemented in com.insync.ems.service.BaseEventHandler

7.25 How to Create an Event Handler

In this section we will discuss how to create and implement new event handler using InSync APIs. Let us create CreatePOHandler

Check the supported java version in the installer document, make sure you use this version only, otherwise it will result in unexpected behaviour

Copy ems.jar, into the build path so that appropriate classes are available for import, you can get these jar files from app/WEB-INF/lib

Step 1. Create a new java class CreatePOHandler , this should extend com.insync.ems.service.BaseEventHandler and the name must end with Handler and the package name should include .handler and should start with com.

Correct package name : com.xxx.yyy.handler
Wrong package name : com.xxx.yyy.something

Correct package name : com.xxx.yyy.handler
Wrong package name : net.xxx.yyy.handler

Import these classes
import com.insync.ems.common.Constants;
import com.insync.ems.event.EventActionParameter;
import com.insync.ems.event.EventContext;
import com.insync.ems.exception.EventParseException;
import com.insync.ems.exception.HandlerFailedException;
import com.insync.ems.service.BaseEventHandler;

Declare the event properties and handler properties using the annotation EventActionParameter as mentioned above. Please note these variables must be public, as they are loaded at runtime using java reflection and hence need access from outside package

/* ---------- event properties ----------*/

```
@EventActionParameter(defaultValue =
Constants.XML_TAGID_ELEMENT,displayName=Constants.DISP_EVENT_TAGID_NAME,
valueMethod="",exampleValue="3034024220649B0000000001")
        public static final String EVENT_TAGID_NAME = Constants.EVENT_TAGID_NAME;


@EventActionParameter(defaultValue = Constants.XML_DEVICE_ELEMENT,
displayName=Constants.DISP_EVENT_DEVICE_NAME,valueMethod="",exampleValue="R1001")
        public static final String EVENT_DEVICE_NAME = Constants.EVENT_DEVICE_NAME;


        @EventActionParameter(defaultValue =
Constants.XML_ANTENNAID_ELEMENT,displayName=Constants.DISP_EVENT_ANTENNA_ID,
valueMethod="",exampleValue="1")
        public static final String EVENT_ANTENNA_ID = Constants.EVENT_ANTENNA_ID;

@EventActionParameter(defaultValue =
Constants.XML_ZONE_ELEMENT,displayName=Constants.DISP_EVENT_ZONE_NAME,
valueMethod="",exampleValue="Pack Zone")
        public static final String EVENT_ZONE_NAME  = Constants.EVENT_ZONE_NAME;

@EventActionParameter(defaultValue = "event_data.po_number",displayName="PO Number",
valueMethod="",exampleValue="PO-123:8989/12")
        public static final String EVENT_PO_NUMBER = "event.po.number";

        @EventActionParameter(defaultValue = "event_data.customer_id",displayName="Customer
ID", valueMethod="",exampleValue="8234567")
        public static final String EVENT_CUSTOMER_ID = "event.customer.id";

        @EventActionParameter(defaultValue =
"event_data.expected_date",displayName="Expected Date",
valueMethod="",exampleValue="12/3/2009 9:34:40 AM")
        public static final String EVENT_EXPECTED_DATE = "event.expected.date";

        @EventActionParameter(defaultValue = "event_data.ship_to_location",displayName="Ship
To",
valueMethod="com.insync.dataobject.service.OrganizationService.getLocationNamesByOrgKey",exa
mpleValue="Fab 3")
        public static final String EVENT_SHIP_TO_LOCATION = "event.ship.to.location";

        @EventActionParameter(defaultValue = "event_data.sold_to_location",displayName="Sold
To",
valueMethod="com.insync.dataobject.service.OrganizationService.getLocationNamesByOrgKey",exa
mpleValue="MOS 8")
        public static final String EVENT_SOLD_TO_LOCATION = "event.sold.to.location";


/*   ----------------  handler properties ----------------------*/


        @EventActionParameter(defaultValue = "YES", displayName="Send
Acknowledgement",valueMethod="com.insync.dataobject.context.entity.trackable.TrackableItem.getTr
ackableValues",exampleValue="Fab 10")
        public static final String HANDLER_SEND_ACKNOWLEDGEMENT =
"handler.send.acknowledgement";

        @EventActionParameter(defaultValue = "YES", displayName="Priority
Order",valueMethod="com.insync.dataobject.context.entity.trackable.TrackableItem.getTrackableValu
es",exampleValue="Fab 9")
```

```
	public static final String HANDLER_PRIORITY_ORDER = "handler.priority.order";

	@EventActionParameter(defaultValue = "MOS6", displayName="Manufacturing
Unit",valueMethod="",exampleValue="MOS6")
	public static final String HANDLER_MANUFACTURING_UNIT = "handler.mfg.unit";
```

Override process(EventContext context) method and implement the business logic here.

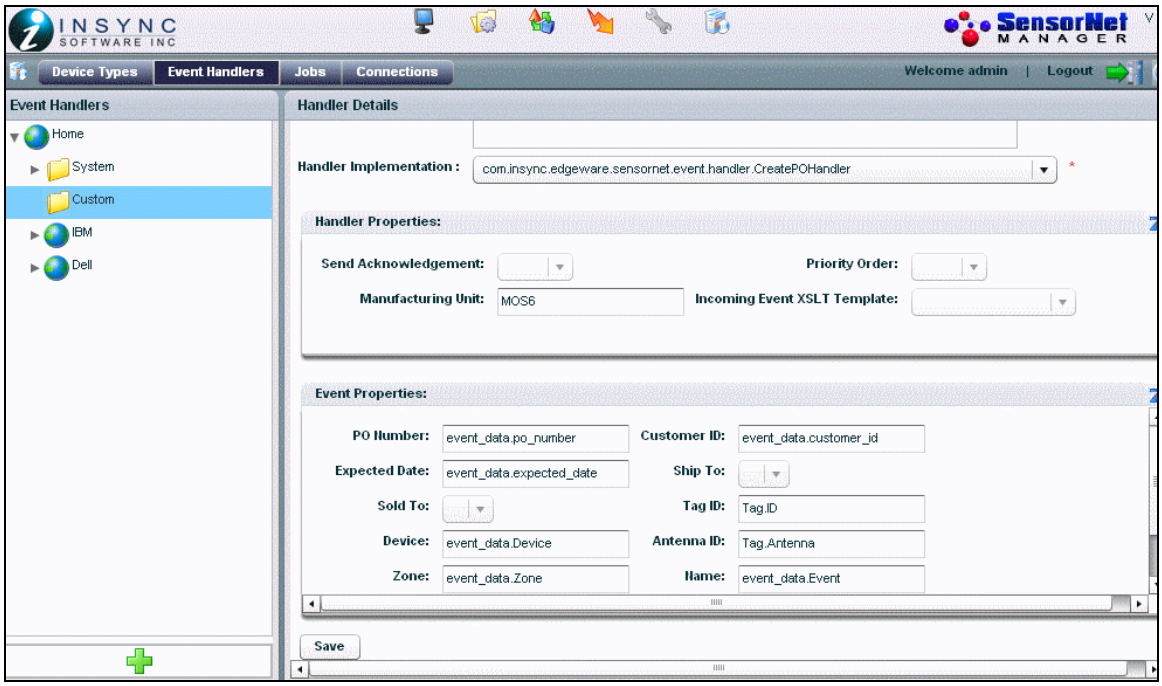
```
	public void process(EventContext context) throws EventParseException {

		// get po_number
		String poNumber =  context.getEventActionDataByName(EVENT_PO_NUMBER);

		// get customer id
		String customerID =  context.getEventActionDataByName(EVENT_CUSTOMER_ID);

		// get expected date
		String expectedDate =
context.getEventActionDataByName(EVENT_EXPECTED_DATE);

		// get shipTo
		String shipTo =
context.getEventActionDataByName(EVENT_SHIP_TO_LOCATION);

		// get soldTo
		String soldTo =
context.getEventActionDataByName(EVENT_SOLD_TO_LOCATION);

		// get send Acknowledgement
		String sendAcknowledgement =
context.getEventActionDataByName(HANDLER_SEND_ACKNOWLEDGEMENT);

		// get Priority Order
		String priorityOrder =
context.getEventActionDataByName(HANDLER_PRIORITY_ORDER);

		// get Manufacturing Unit
		String mfgUnit =
context.getEventActionDataByName(HANDLER_MANUFACTURING_UNIT);

		//to-do.  validate the data and create PO
	}
```

Step 2. Compile the above class and create a jar file and copy to the class path . For tomcat, it is

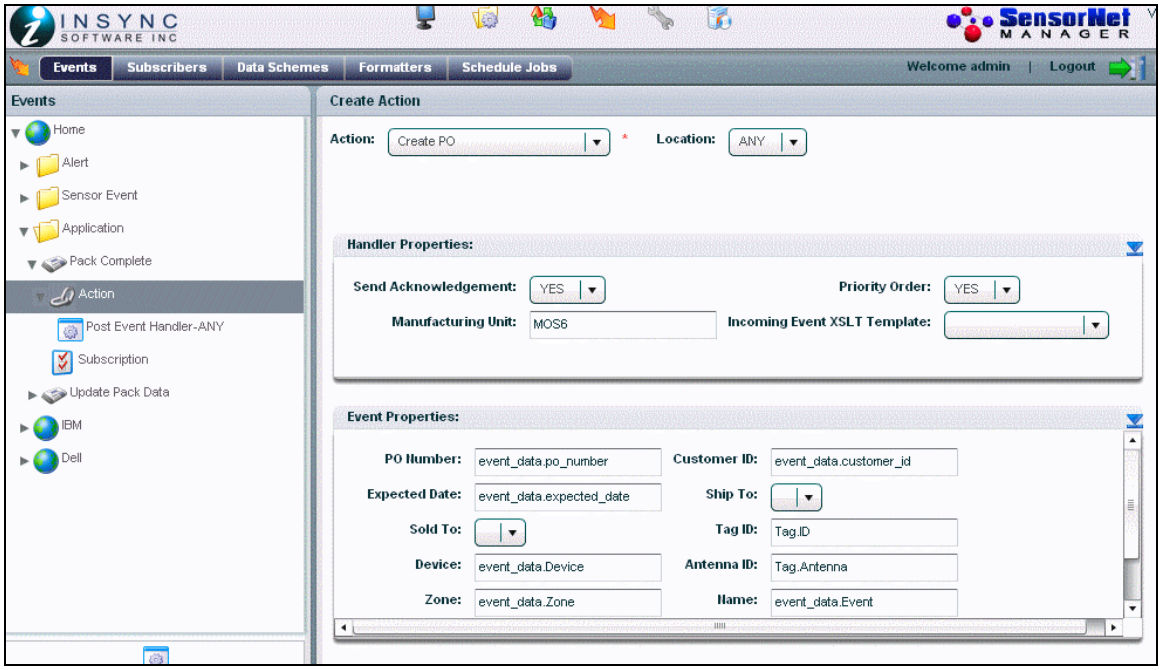
```
		$EDGEWARE_INSTALLATION_HOME\sensornet\app\WEB-INF\lib
```

Step 3. Restart the server. Login to SensorNet, go to Software → Event Handler screen.

Click on Custom and Add at bottom to create a new handler, select this handler implementation class from the drop down of Handler Implementation – you should see your new class here. Save the handler after entering a name for this handler.



Step 4. Create action on this handler, go to Event Management → Events and browse to the event on which you want create this action, then click on Create Action button, select the handler you have selected in step 3 from Action drop-down, at this stage you can edit the values if you need and click on save



Step 5. Create and post the event mentioned in Step 4, your handler implementation will be invoked and the code specified in process method will be executed

Complete code of the example is attached here

CreatePOHandler.java

7.26 List of ValueMethods

We have the following list of ValueMethods that you can reuse. The list is categorized according to Services for eg: AdminService , Material

AdministrationService

Value-Provider Methods	Purpose
com.insync.web.flex.service.AdministrationService.getUserNamesByOrgKey	This value-provider method renders drop down of User Names who are accessible to current organization.
com.insync.web.flex.service.AdministrationService.getOrgTypesByOrgKey	This value-provider method renders drop down of Organization types exists in current organization.
com.insync.web.flex.service.AdministrationService.getUserTypesByOrgKey	This value-provider method renders drop down of user types exists in current organization.
com.insync.web.flex.service.AdministrationService.getOrgsBySaasOrgKey	This value-provider method render drop down of names of organization exists in the current enterprise.
com.insync.web.flex.service.AdministrationService.getRolesByOrgKey	This value-provider method renders drop down of roles names exists in current organization.
com.insync.web.flex.service.AdministrationService.getAllUsersInEnterpriseByOrgKey	This value-provider method renders drop down list of all the user name exists in current organization.

AlertInformation

Value-Provider Methods	Purpose
com.insync.dataobject.context.entity.report.AlertInf	This value-provider method renders drop down of types of UI those are INFORMATIONAL,

ormation.getTypesForUI	ACTIONABLE.

Application

Value-Provider Methods	Purpose
com.insync.dataobject.context.entity.type.Application.getApplicationNamesByOrgKey	This value-provider method renders drop down of application names exists in current organization.

BusinessDocument

Value-Provider Methods	Purpose
com.insync.dataobject.context.entity.doc.BusinessDocument.getBusinessDocumentsByOrgKey	This value-provider method renders drop down of Business documents names exists in current organization.

CatalogService

Value-Provider Methods	Purpose
com.insync.dataobject.framework.catalog.CatalogService.getCategoriesOfContactTypeByPipeSeperated	This value-provider methods renders drop down of Categories exists in the “orgname_ContactType” catalog of current organization.

ConnectionService

Value-Provider Methods	Purpose
com.insync.web.flex.service.ConnectionService.getEmailConnectionNamesByOrgKey	This value-provider method renders drop down of names of the Email connections exists in current organization.

EventExplorerUIService

Value-Provider Methods	Purpose
com.insync.web.service.EventExplorerUIService.getFormatterByOrgKey	This value-provider method renders drop down of FreeMarker-based Formatters exists in current organization.
com.insync.web.service.EventExplorerUIService.getEventNamesAndEmptyEventByOrgKey	This value-provider method renders drop down of event names, which are not Alert Events, exists in current organization.
com.insync.web.service.EventExplorerUIService.getEventNamesByOrgKey	This value-provider method renders drop down Event Names, which are not Alert Events, in current organization.
com.insync.web.service.EventExplorerUIService.getScheduleNamesByOrgKey	This value-provider method renders drop down of Schedule Names exists in current organization.
com.insync.web.service.EventExplorerUIService.getTriggerTimeUnits	This value-provider methods renders drop down of Time Units which are SECOND, MINUTE, HOUR , DAY , WEEK , MONTH
com.insync.web.service.EventExplorerUIService.getEventIdNamesByOrgKey	This value-provider method renders the drop down with the event ID , event name pair in the following way Event ID=Event Name
com.insync.web.service.EventExplorerUIService.getFormatterNamesByOrgKey	This value-provider method renders drop down of XSLT-based Formatter of type exists in current organization.

Material

Value-Provider Methods	Purpose
com.insync.edgeware.context.entity.type.Material.getMaterialNumbersByOrgKey	This value-provider method renders drop down of Material Numbers exists in current organization.
com.insync.edgeware.context.entity.type.Material.getMaterialTypesByOrgKey	This value-provider methods renders drop down of Material Types exists in current

	organization.

OrganizationService

Value-Provider Methods	Purpose
com.insync.dataobject.service.OrganizationService .getLocationNamesByOrgKey	This value-provider method renders the drop down of all the location names of the organization
com.insync.dataobject.service.OrganizationService .getCustomerNamesByOrgKey	This value-provider methods renders drop down of Customer Names exists in current organization.
com.insync.dataobject.service.OrganizationService .getSupplierNamesByOrgKey	This value-provider methods renders drop down of Supplier Names exists in current organization.
com.insync.dataobject.service.OrganizationService .getZoneNamesByOrgKey	This value-provider methods renders drop down of zone names exists in current organization.

PrintService

Value-Provider Methods	Purpose
com.insync.edgeware.sensornet.printer.service.PrintService.buildPrinterNamesForHandler	This value-provider methods renders drop down of Printer Names exists in current organization.
com.insync.edgeware.sensornet.printer.service.PrintService.getPrintDocNamesForHandler	This value-provider methods renders drop down of Printer Document names exists in current organization.

SensorNetService

Value-Provider Methods	Purpose

com.insync.edgeware.sensornet.savant.SensorNetService.getDeviceNamesByOrgKey	This value-provider method renders drop down of Device Names exists in current organization.

TrackableItem

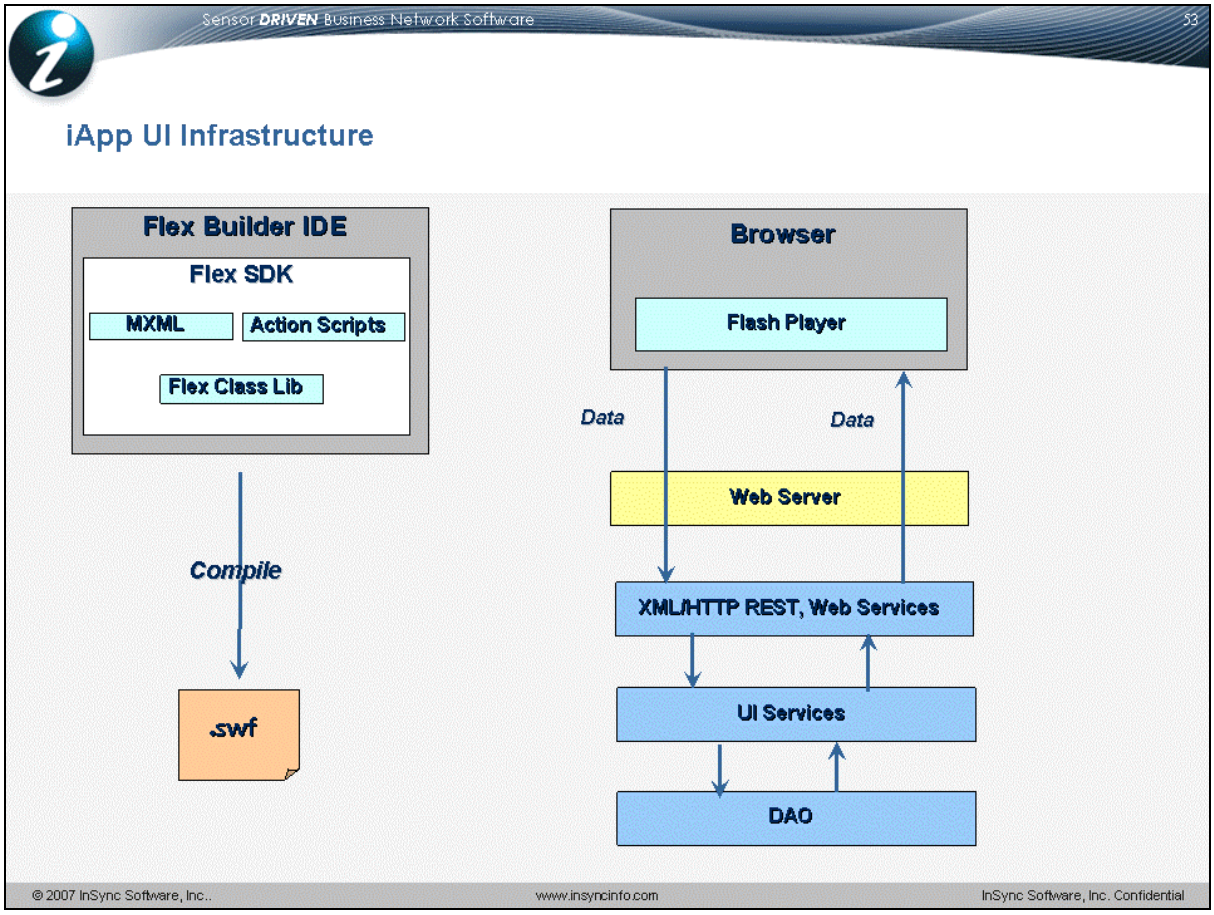
Value-Provider Methods	Purpose
com.insync.dataobject.context.entity.trackable.TrackableItem.getTrackableItemUniqueKeyValues	This value-provider method renders drop down of Unique ID of trackable Items those are tagId, assetId, alternateTagId, systemId.
com.insync.dataobject.context.entity.trackable.TrackableItem.getTrackableValues	This value-provider method renders drop down of YES or NO Flag. If selected YES it possibly brings all the trackable values.
com.insync.dataobject.context.entity.trackable.TrackableItem.getTrackableModeValues	This value-provider method renders drop down with trackable Modes those are onChange and Always.
com.insync.dataobject.context.entity.trackable.TrackableItem.getValidInventoryActions	This value-provider method renders the drop down of inventory actions those are RECV, MOVE, and SHIP.

8 iApp UI Services

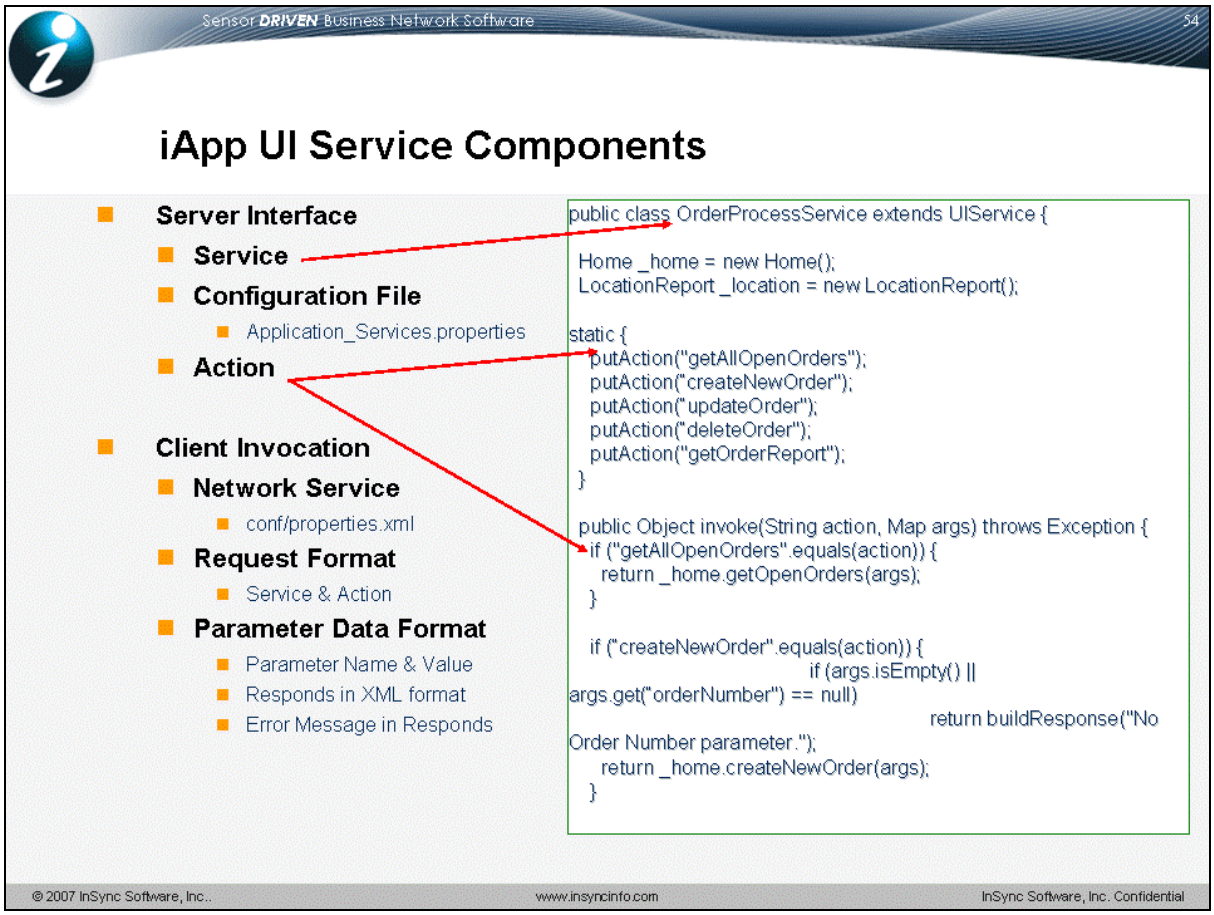
8.1 Introduction

This document describes the UI Services used in iApp. UI Services are server side components that provide data to the UI screens. All UI screens call the UI Service to get the data and render the screen.

8.2 Infrastructure



As you can see from above diagram, all the flex calls will come to BaseGatewayServlet through HTTP REST method , BaseGatewayServlet in turn will route the request to appropriate UI Service and invoke the action mentioned in the request. UI Service will execute the action method and create a response object and send it back to the flex client. The request, response objects are in xml format



Every UIService has 3 components

8.3 ServiceName

This is the name used by flex client to invoke the service class. We do the mapping of the service name to the class name in WEB-INF/classes/Application_Services.properties file. Example is

OrderProcessService=com.insync.edgeware.sensornet.web.service.OrderProcessService

8.4 Action

Action corresponds to a method in the service class, it executes one logical unit of work. Examples : createOrder, deleteOrder, updateOrder, getAllOpenOrders. Each action is registered using putAction(actionName) in the service static block. Action has one input parameter Map which will have all the parameters. Return type is String with the response in xml format

8.5 Parameters

Every action has input parameter Map with the parameters. Parameter value could be a xml

8.5.1 Examples : Creating UIService

1. We need to create a class OrderUIService in any package (suppose here package name is “test.com.insync.edgeware.web.service “) that extends com.insync.web.flex.service.UIService
2.
3. It should have static block where all the actions are registered. Example

```
static {
putAction("getAllOpenOrders");
putAction("createNewOrder");
putAction("updateOrder");
putAction("deleteOrder");
}
```

4. We need to override invoke method and call these actions

```
public Object invoke(String action, java.util.Map args) throws Exception {

Object response = "";
if ("getAllOpenOrders".equals(action))
response = getAllOpenOrders(args);
else if ("createNewOrder".equals(action))
response = createNewOrder(args);
else if ("updateOrder".equals(action))
response = updateOrder(args);
else if ("deleteOrder".equals(action))
response = deleteOrder(args);

return response;

   }
```

5. Now we need to create these action methods and implement the business logic

```
public String getAllOpenOrders(Map map) throws Exception {

return buildResponse("Returns all Open Orders");
}

public String createNewOrder (java.util.Map map) throws Exception {

// retrieve order number
String orderNumber=null;
if (map.get("orderNumber") != null) {
orderNumber = ((String[]) map.get("orderNumber"))[0];
}

if(orderNumber==null || orderNumber.length()==0){
return buildResponse("Order Number is required");
}

return buildResponse("Created new Order");
}

public String updateOrder(java.util.Map args) throws Exception {
```

```
		return buildResponse("Updates Order");
	}

	public String deleteOrder(java.util.Map args) throws Exception {

		return buildResponse("Deletes Order");
	}
```

In this example orderNumber is the parameter name used in flex client

1. Response object

The structure of the response object should be like this

```
<?xml version=\"1.0\"?>
<Response>
<Status>success/failed</Status>
<Description>Location with this name[Fab3] does not exist </Description>
<Data>
……
……
</Data>
</Response>
```

Description is mandatory for failed messages, we need to give user friendly messages here so that it's displayed in UI to the user.

Data element can have its own structure

When you call the method "getAllOpenOrders" the response object will be like this

```
<Response>
	<Status>success</Status>
	<Result>
		<Orders>
			<Order />
		</Orders>
	</Result>
</Response>
```

When you call the method "createNewOrder" the response object will be like this

```
<Response>
			<Status>failed</Status>
			<Description>Order Number is required</Description>
</Response>
```

When you call the method "updateOrder" the response object will be like this

```
<Response>
			<Status>success</Status>
			<Result>Order updated successfully</Result>
</Response>
```

When you call the method “deleteOrder” the response object will be like this
<Response>
<Status>success</Status>
<Result>Order deleted successfully</Result>
</Response>

2. All input,output parameters must have one parameter called OrgKey

3. Add created class in WEB-INF/classes/Application_services.properties file like this with the package name,

OrderService= test.com.insync.edgeware.web.service.OrderUIService

Make sure the “OrderService” name is not given to any other service this file.

4. Base class UIService has some utility methods that all UI Services can use

protected Document createJdomDocument(InputStream dataStream)
throws JDOMException, IOException

this can be used to create Jdom object from the input parameters

protected String exportDocument(org.w3c.dom.Document doc)
throws IOException

this can be used to create xml string from Document object

5. 9. Create the jar file for above created class and place the created jar in the /WEB-INF/lib folder, and make sure that above created jar file name is unique across the /WEB-INF/lib folder.

6. 10. Start the application.

Complete java file is below.

```
package com.insync.edgeware.web.service;

/**
 * This class is used Create new order, update existing order,
 * deleting the existing order and retrieving all open orders.
 *
 */
public class OrderUIService extends com.insync.web.flex.service.UIService {

	/**
```

```
	 * Each action defined should be registered with following static block.
	 * Ex:- If new action is created with the name "approveOrder", it should
	 * be registered with this class in static block using putAction("approveOrder")
	 */
	static {
		putAction("getAllOpenOrders");
		putAction("createNewOrder");
		putAction("updateOrder");
		putAction("deleteOrder");
	}

	/**
	 * This is method is used invoke the particular action based
	 * on the parameter action.
	 * Ex:- If parameter action comes with value "updateOrder" from flex UI then
	 * "updateOrder" method will be invoked in this method.
	 */
	public Object invoke(String action, java.util.Map args) throws Exception {

		Object response = "";

		if ("getAllOpenOrders".equals(action))
			response = getAllOpenOrders(args);
		else if ("createNewOrder".equals(action))
			response = createNewOrder(args);
		else if ("updateOrder".equals(action))
			response = updateOrder(args);
		else if ("deleteOrder".equals(action))
			response = deleteOrder(args);

		return response;
	}

		/**
		 * Retrieves all open orders and build the usable XML, one of the
response xml form is
		 * <Orders>
		 *		<Order>
		 *			<id></id>
		 *			<date></date>
		 *		</Order>
		 *		<Order>
		 *			...
		 *		</Order>
		 *		.....
		 * </Orders>
		 * @param args
		 * @return
		 * @throws Exception
```

```
	 */
	public String getAllOpenOrders(java.util.Map args) throws Exception {
		//TODO - code for retrieving the orders. If orders exists this code gives data
		/// in xml form (one of the mentioned form is in above JavaDoc)
		//Data ->  <Orders>
		//		<Order>
		//			<id></id>
		//			<date></date>
		//		</Order>
		//		<Order>
		//			...
		//		</Order>
		//		.....
		//  </Orders>
		String data = "<Orders><Order></Order></Orders>"; // if order exists this will be one of the response xml form.
		return buildResponse("success", data);
	}

	/**
	 * This method is used to create new order.
	 * After successful creation it returns success message as "Create order Successfully"
	 * @param map
	 * @return
	 * @throws Exception
	 */
	public String createNewOrder (java.util.Map map) throws Exception {

		// retrieve order number
		String orderNumber=null;
		if (map.get("orderNumber") != null) {
		orderNumber = ((String[]) map.get("orderNumber"))[0];
		}

		if(orderNumber==null || orderNumber.length()==0){
			return buildFailedResponse("Order Number is required");
		}

		return buildResponse("success", "Created order Successfully");

	}

	/**
	 * This method is used to update the existing order.
	 * After successful updation it returns success message as "Updated Order successfully".
	 *
```

```
     * @param args
     * @return
     * @throws Exception
     */
    public String updateOrder(java.util.Map args) throws Exception {

        //TODO - Code to update the mentioned order.
        return buildResponse("success", "Order updated successfully");
    }

    /**
     * This method is used to delete the existing order.
     * After successful deletion it returns success message as "Updated
Order successfully".
     *
     * @param args
     * @return
     * @throws Exception
     */
    public String deleteOrder(java.util.Map args) throws Exception {

        //TODO - Code to delete the mentioned order.
        return buildResponse("success", "Order deleted successfully");
    }
}
```

---

9 Image Map Calibration

9.1 Image Calibration

Image Calibration is a process used to determine X/Y/Z (Longitude/Latitude/Altitude) coordinate values for any pixel of the given image after calibration. iApp System supports calibration of images based on the X/Y/Z coordinate system or Longitude/Latitude/Altitude values. For calibrating the image at least 2 points with corresponding coordinate values should be given.

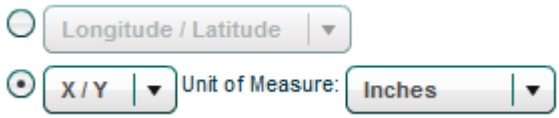
From these points and their values, various ratios will be calculated and those ratios are used to determine coordinate system value for any given image point.

To get accurate results, following points have to be followed.

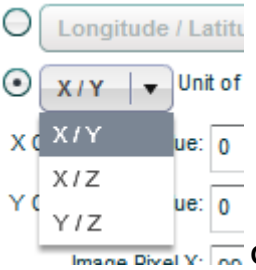
1. The more distance between the points, the possibility of accurate result is more. So, diagonal points or near corner points are preferred.
2. If the given points fall on the same axis (X/Y/Z), then more than 2 points are required.
3. If the system is not able to calculate the ratios by using the given points then the system will ask for more points.
4. The angle between X and Y axis should be 90° when calculated in counter clock wise direction.
5. If the calibration is of type Longitude/Latitude/Altitude then North should point to top ( ).
6. For out door images, satellite images are preferred. Or picture should be taken from top. Picture taken from side will not give proper results.
7. Images used for calibration with Longitude & Latitude values can not be rotated.
8. Rotated images are supported with X/Y calibration but it is always better not to rotate for accurate results.

9.2 Calibration Process:

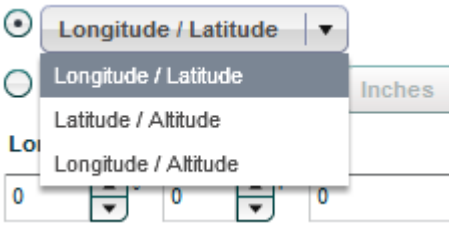
1. Select the calibration type whether it is based on X/Y/Z (used usually for in door images) values or based on the Longitude, Latitude, Altitude (used usually for out door images) values. This can be done by selecting radio button as shown below.



2. Select the coordinate system by choosing a value from the combo box. If the calibration is based on X/Y/Z then the possible value is X/Y or X/Z or Y/Z. If the calibration type is based on Longitude/Latitude/Altitude then the possible value is Longitude/Latitude or Longitude/Altitude or Latitude/Altitude.



Or



3. Adding a Point:

   a. Move the mouse cursor on the image and you will see a pointer on the image and this pointer will be moving with mouse cursor.

   b. Clicking on the image will fix this pointer and then onwards this pointer can be moved by dragging.

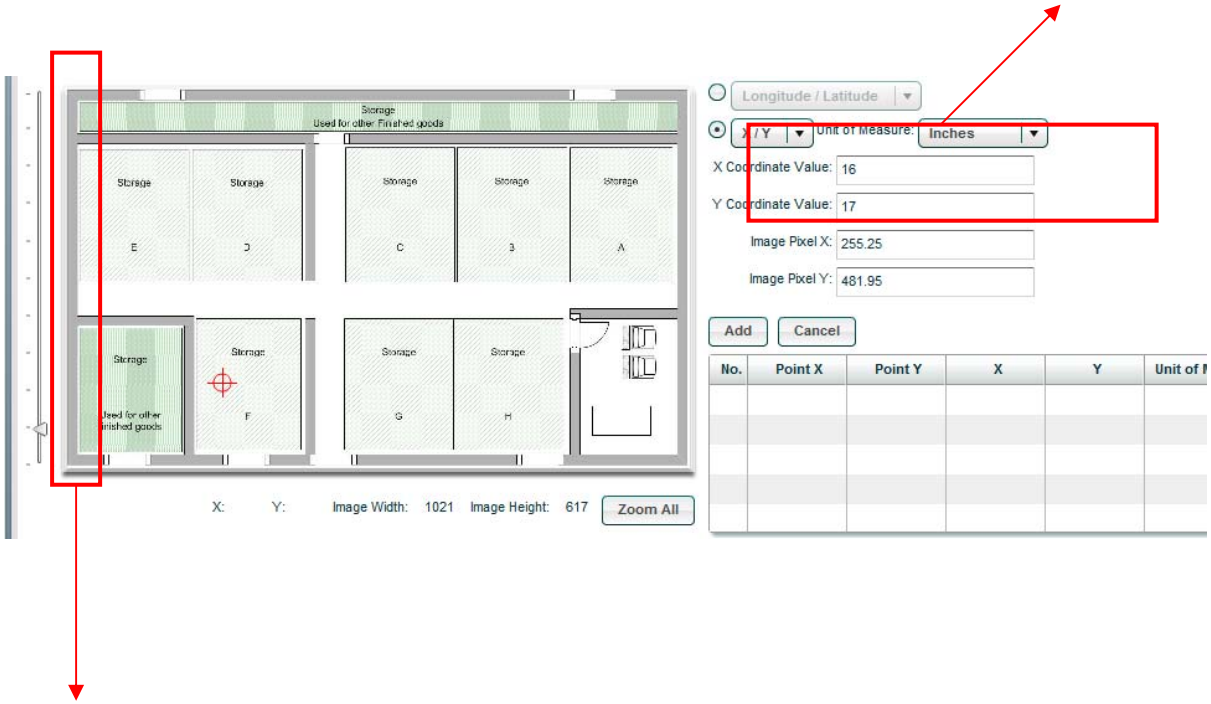
   c. The center of the pointer should be placed on the image at the point whose coordinate values are known.

   d. The pointer on the image can be set precisely by zooming the image and moving the pointer by dragging it. Image zooming can be done using the slider which is available on the left of the image.

   e. The pointer on the image can be changed by changing the values of “Image Pixel X” or “Image Pixel Y” or both. Using these values the pointer can be move very precisely.

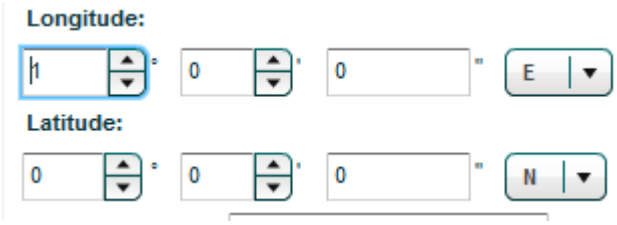
The coordinate system values should be given on the text fields available next to the image.

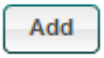
Coordinate System Values



Zoom Slider

If the calibration is based on the Longitude/Latitude then the value should be given in degrees (°), minutes (') & seconds (") as shown below.

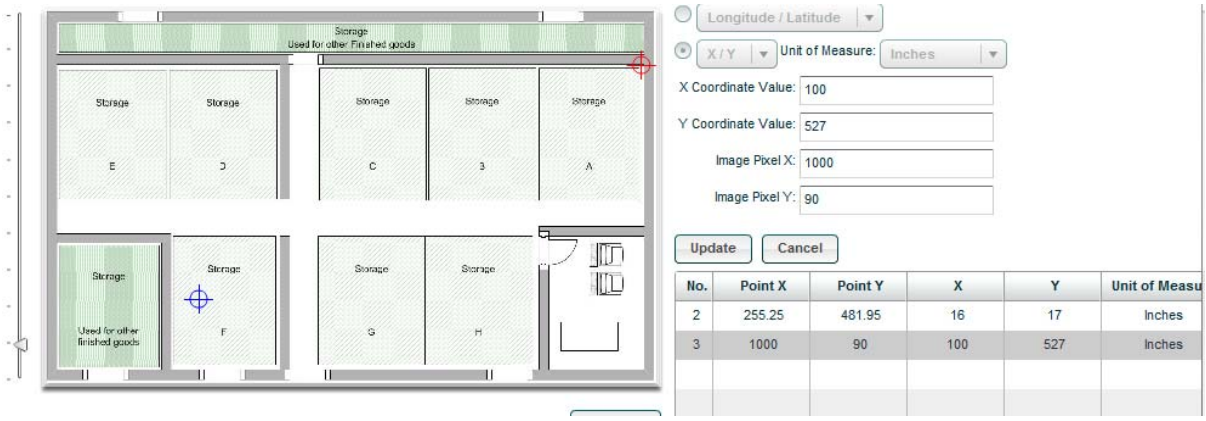


Once the values have been given add the point by clicking on button Add (

). This will make the pointer color to blue.

Now one more pointer can be added by following the steps given above from step a to step d.

1. Editing a Point:
   a. Select the point to edit by selecting the row in data grid.
   b. Click on edit button available below the grid. This will change the color corresponding pointer to red.
   c. Pointer can be moved by dragging it and values can be edited in the text fields.
   d. Update the values by clicking on the button Update.



2. Save the calibration by click on the Save Calibration ( ) button.

3. After successful save calibration can be validated by click on the button Validate. This will make the screen to work in validate mode. In this mode, when the mouse cursor is moved on the image corresponding coordinate values will be shown below the image.
4. Clicking on the Edit Calibration button will change the screen to edit mode.
5. “Zoom All” button resets the zoom level to one.

9.3 Image Regions

Usually a single image represents lot of physical area and this area will be divided into various regions for administration purposes. These regions will have names to refer. To replicate the same functionality on image, image has to be divided into separate regions with names. These regions can be with various shapes.

Using this screen, image can be divided into various regions with names and properties. Regions can be created with following shapes.

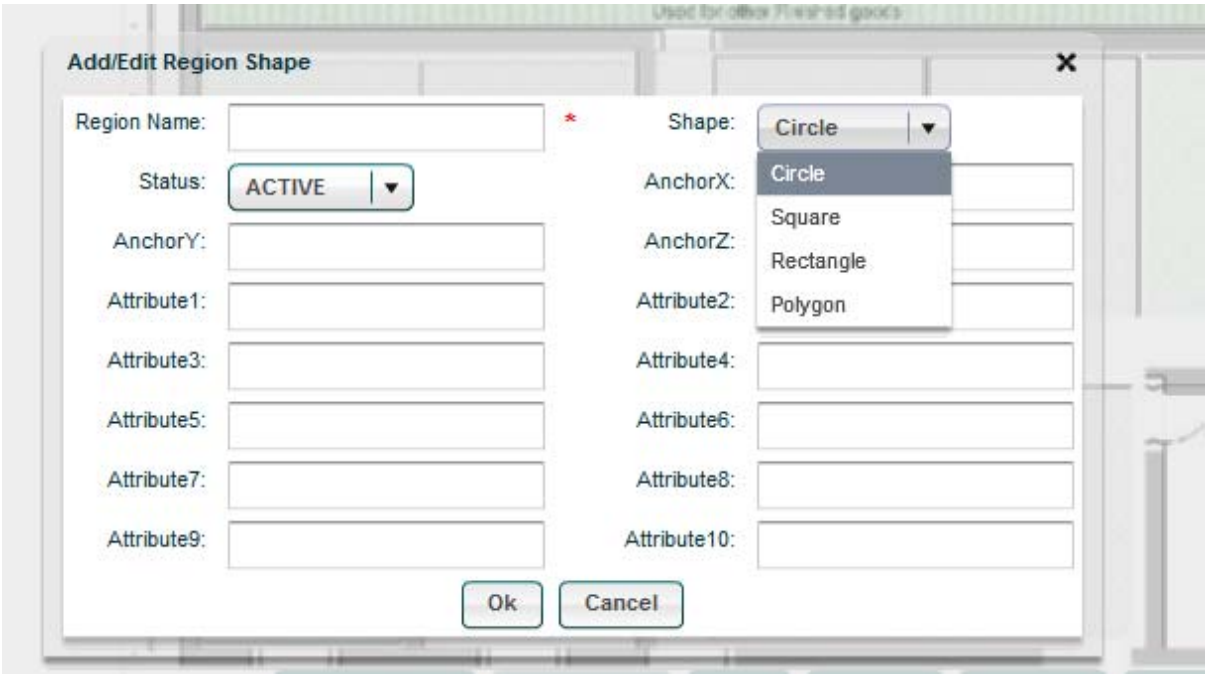
- Circle
- Square
- Rectangle
- Polygon.

Once the region is defined, region can be resized by moving the corners of the region. Region can be moved by dragging it.

Image picture can be zoomed using the slider bar and clicking on the “Zoom All” button will reset the zoom to 1. Even after zooming in the image, regions can be moved and resized. Zooming in the image and resizing the region will make drawing the region more precisely.

9.3.1 Defining a Region:

1. Click on the “Define Region” button. This will open up a pop up and the popup will look like as given below.



2. Enter the name of region and select the shape of the region. Once defining the region is over, shape can not be changed. So care has to be taken before selecting the shape.
3. The value of the properties of AnchorX, AnchorY and AnchorZ captures the point, which is taken as the anchor point for the region. This should be inside of the region to make sense when used any of the application.
4. Attributes1 to Attribute10 can be used to save any custom properties related to that region.
5. Click button “Ok” to draw the region. Now a message will be shown on how to draw the region. The shapes can be drawn as described below.
    a. Circle: Click on the image to select the center of the circle and drag the mouse to draw the circle. As the mouse moves, a circular region can be seen changing its circular area. Once the desired area is covered, release the mouse and defining the circle is over.

    b. Square: Click on the image to select any corner of the square and drag the mouse to draw the square. As the mouse moves, a region in square shape can be seen and changes the area as per the mouse movement. Release the mouse to complete the defining the region.

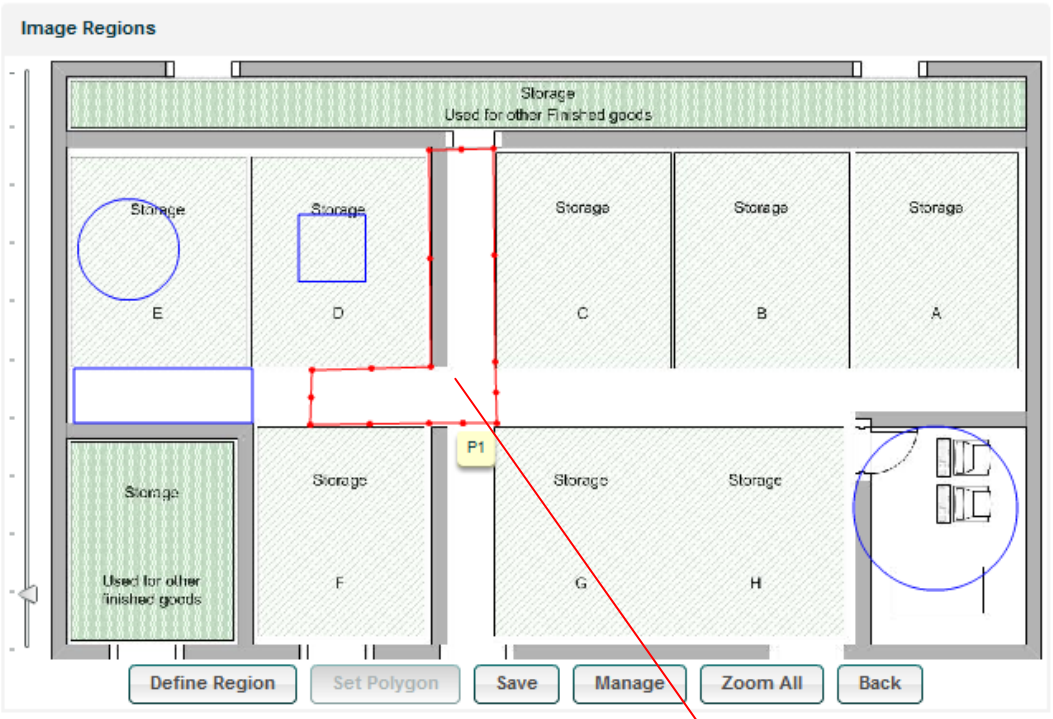
    c. Rectangle: Click on the image to select any corner of the rectangle and drag the mouse to draw the rectangle. As the mouse moves, a region in rectangle shape can be seen and changes the area as per the mouse movement. Release the mouse to complete the defining the region.

    d. Polygon: Defining the polygon is different from defining other shapes. Click on the image in sequence to form the shape of polygon. Here each click refers to a corner point of the polygon. After each click, the polygon is drawn based on the defined corners. Once the polygon has been drawn as required, click on “Set Polygon” button to complete the process of defining the polygon.

6. Clicking on the Save button saves all the regions.

9.3.2 Moving a Region:

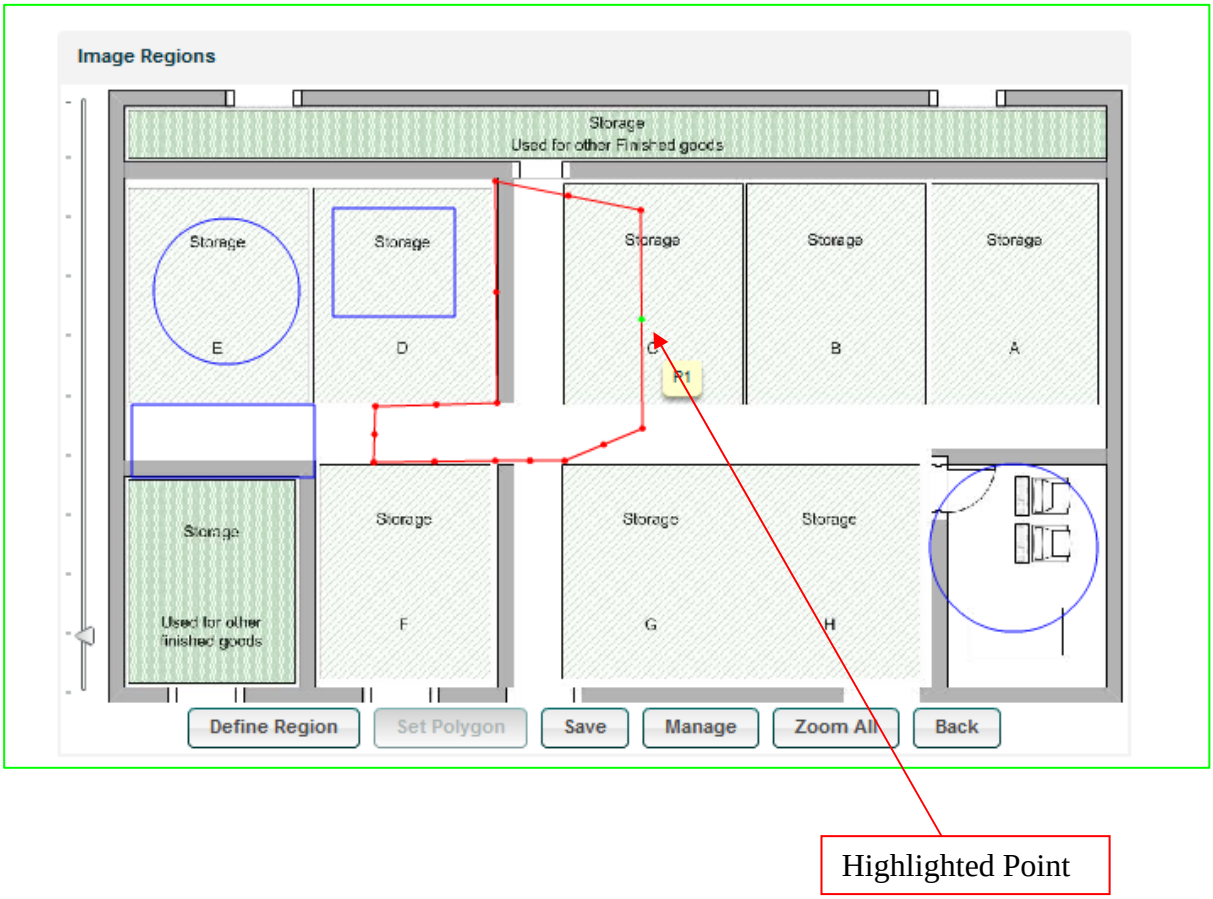
By default the regions borders are in blue color. When mouse is moved over any region, the corresponding region gets highlighted and the border color will be changed to red color. So region can be selected by moving the mouse over by a region. Once the mouse moves out of the region, the region will be deselected automatically and border color will be changed back to blue. When the region is in highlighted mode, region can be moved by dragging the region (click on the region, hold it and move the mouse).



To move region by few pixels, it is better to zoom the image first and then select the region and move it.

9.3.3 Resizing a Region:

When the region is highlighted, all the corner points and middle points of that line can be seen as dots. For circle, center point and four points (one each on each side) can be seen as dots. These points (dots) are used to resize the region. When the mouse moves to any of these points, the corresponding point get highlighted by and color of that changes to green.



When the color of the point is green, region can be resized by dragging the point slowly. At this point of time behavior/drawing of the region changes based on its shape. When the region is getting resized the behavior will be describe below.

Circle Shape: When any corner point is selected and dragged, then the whole circle expands/contracts based on the drag is outwards/inwards.

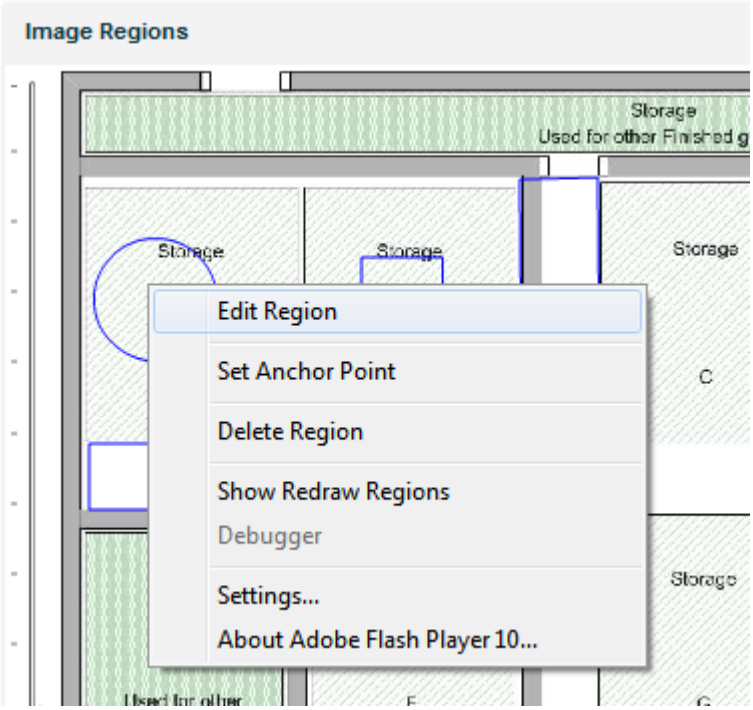
Square Shape: When any corner point is selected and dragged, the diagonal point will remain at its position and all other 3 corner points will move outwards/inwards based on the drag direction. As the region shape is square, all the side lengths will be maintained to same length.

Rectangle Shape: When any corner point is selected and dragged, the diagonal point will remain at its position and all other 3 corner points will move outwards/inwards based on the drag direction. When any of the line middle point is selected and dragged, then the adjacent 2 points will be moved towards drag direction.

Polygon Shape: When any corner point is selected and dragged, the all other points will remain static at their positions and the selected will move towards the drag direction and the lines joined to this point will be redrawn as the move progresses. When any of the line middle point is selected and dragged, then the adjacent 2 points will be moved towards drag direction and all other points will remain static at the position, The lines joined these adjacent points will be redrawn as the move progresses.

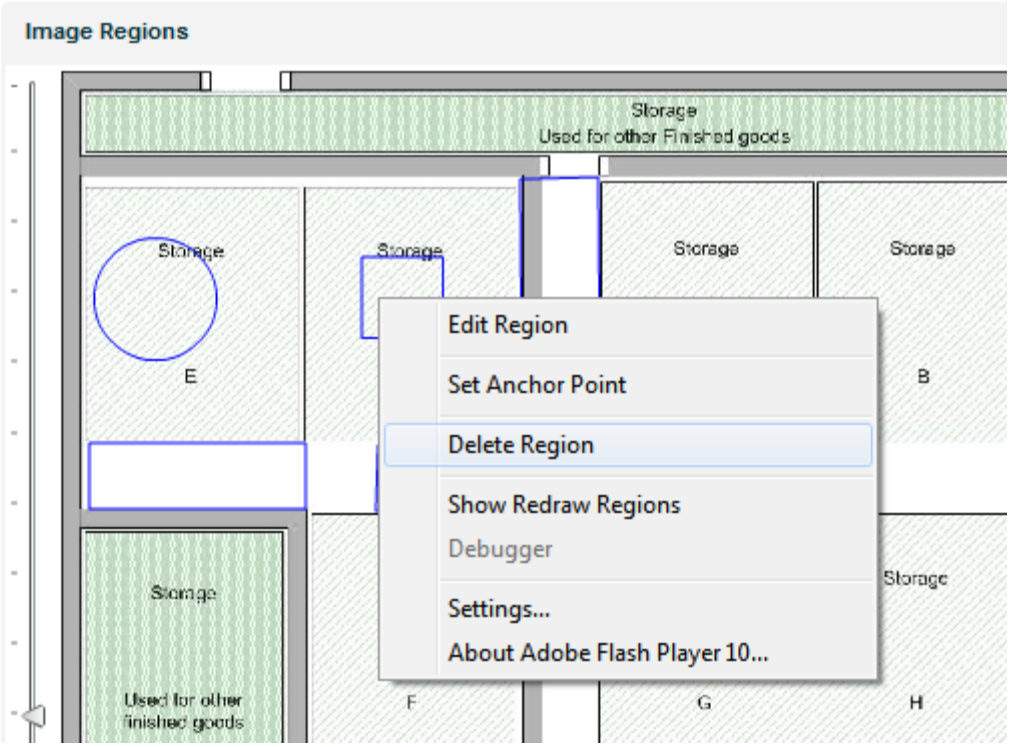
9.3.4 Editing a Region:

Select the region to be edited and select “Edit Region” menu item in the context menu. Mouse right click will bring up the context menu. This will bring the popup screen where all the properties can be edited except the shape.



9.3.5 Deleting a Region:

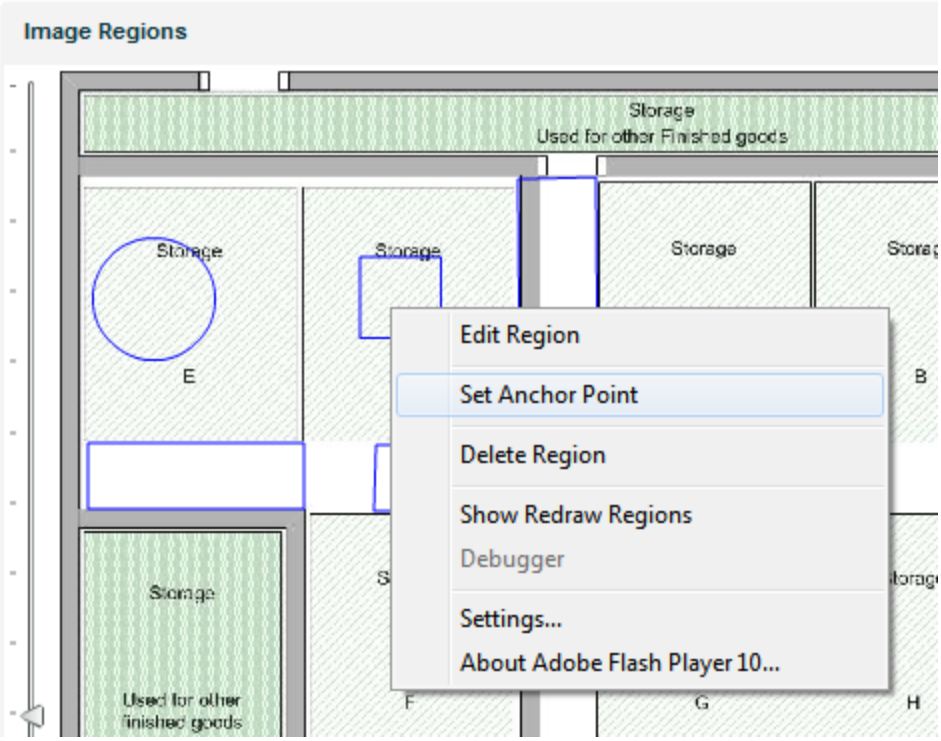
Select the region to be edited and select “Delete Region” menu item in the context menu. Mouse right click will bring up the context menu. This action will delete the selected region from this session. To delete the region permanently, “Save” button has to be clicked.



9.3.6 Setting Anchor Point:

Anchor point is the anchoring point in the region, which is important point in the region. This point is used to show any icon on the region.

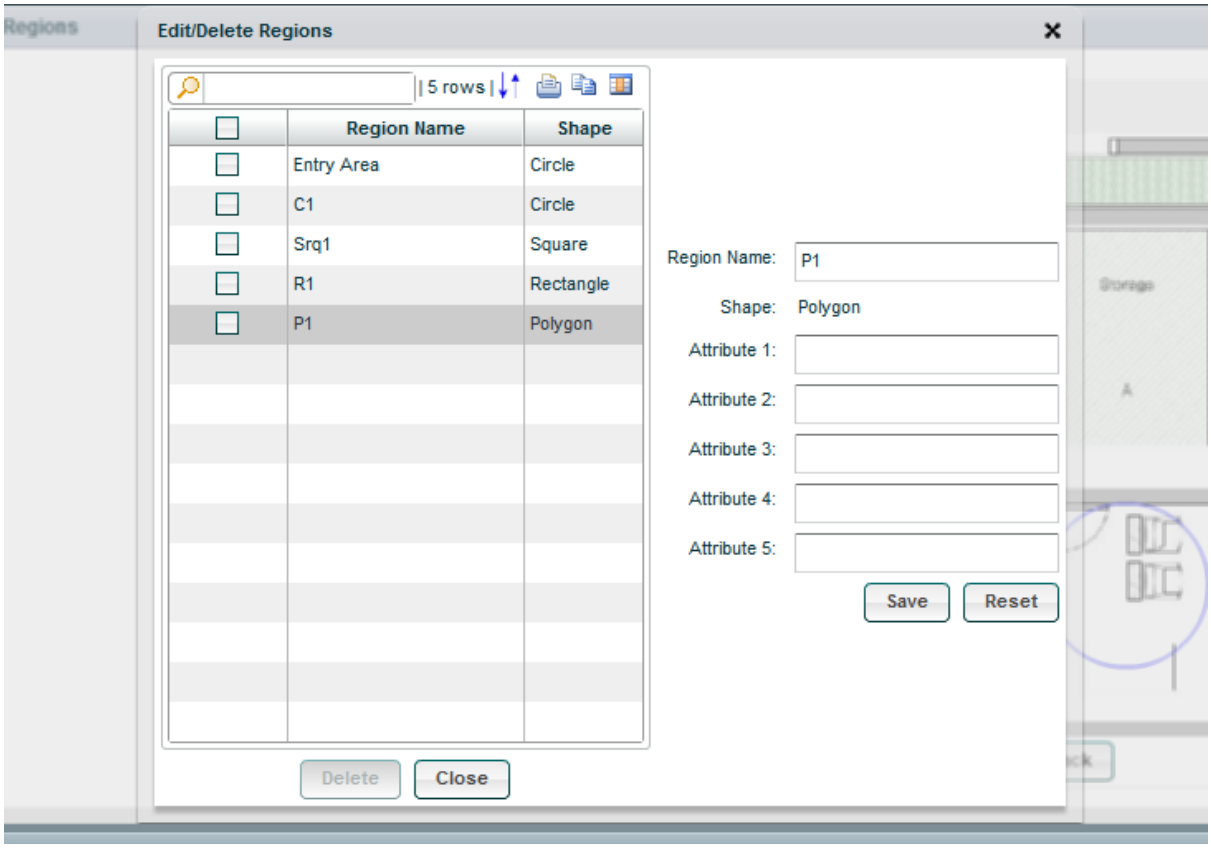
For setting anchoring point to a region, select the region and then choose the “Set Anchor Point” from the context menu. There will be a message (“Please click on Region to set the Anchor Point”) asking to choose anchoring point. Now click on the region where the anchoring point has to be set and message will be show saying anchoring point has been set.



9.3.7 Managing Regions:

Clicking on the “Manage” button will open up a pop up window. From this window more than one region can be deleted. Regions can be updated.

The window has a data grid (table), in which each region is shown as a row. Selecting a row will show its details on the right side and the region details can be updated. Selecting the check boxes and then clicking on the delete button will delete the checked the regions.



9.4 Locator Tags

Locator tags are RFID tags with X/Y/Z or Longitude/Latitude/Altitude values. This tag will have the flag Locator Tag set to true.

8.4.1 Creating a Locator Tag:

1. Move the mouse cursor on the image and you will see a pointer on the image and this pointer will be moving with mouse cursor.

2. Click on the image where the Tag is currently placed or available. This will stick the point to the image. Now the position of the pointer can be adjusted by dragging it.

3. If exact place of the tag is known with X/Y/Z values then those can be given on the right side. Updating these values will update the tag position on the image also.

4. Enter the Tag ID value of the tag on the right side.

5. Click on “Add” button to add this icon to the data grid table and this will make the corresponding tag pointer to static and the pointer color is changed to blue.

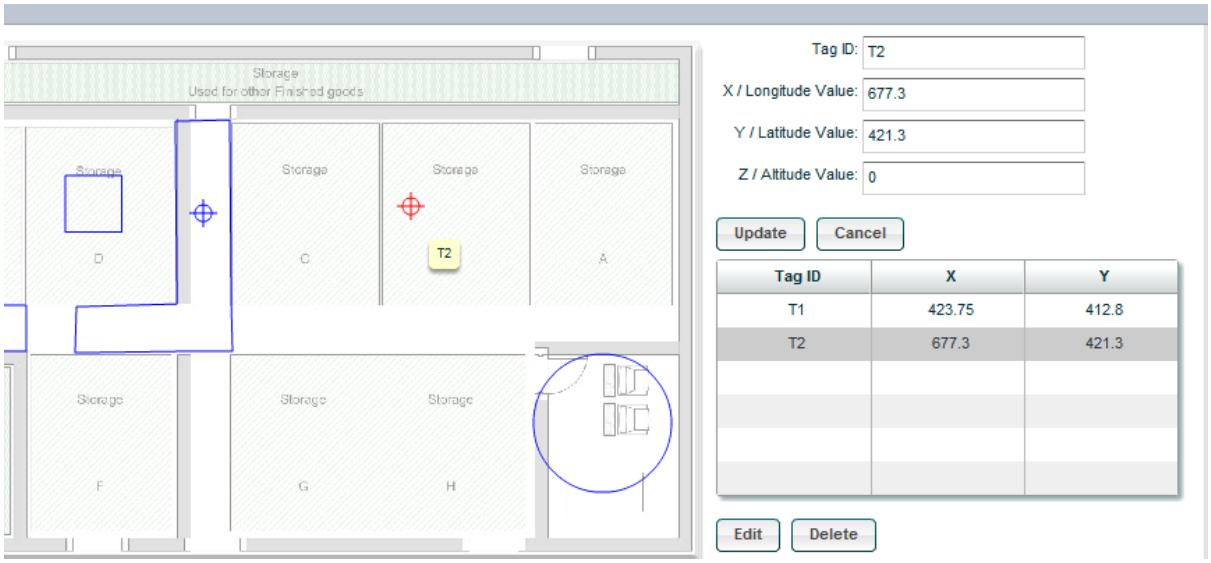
6. Now move the mouse over the image and follow the above steps to add another locator tag.

7. Click on “Save” button will save all the locator tags.

8.4.2 Updating Locator Tag:

1. Select the locator tag to be updated in the data grid table and click on “Edit” button.

2. The color of the corresponding pointer is turned to red and now this pointer can be moved by dragging it or changing the values of X/Y/Z.



a. If needed Tag ID value also can be changed.
b. Click on “Update button to update the value to the data grid.
c. Clicking on “Save” button will save all the locator tags.

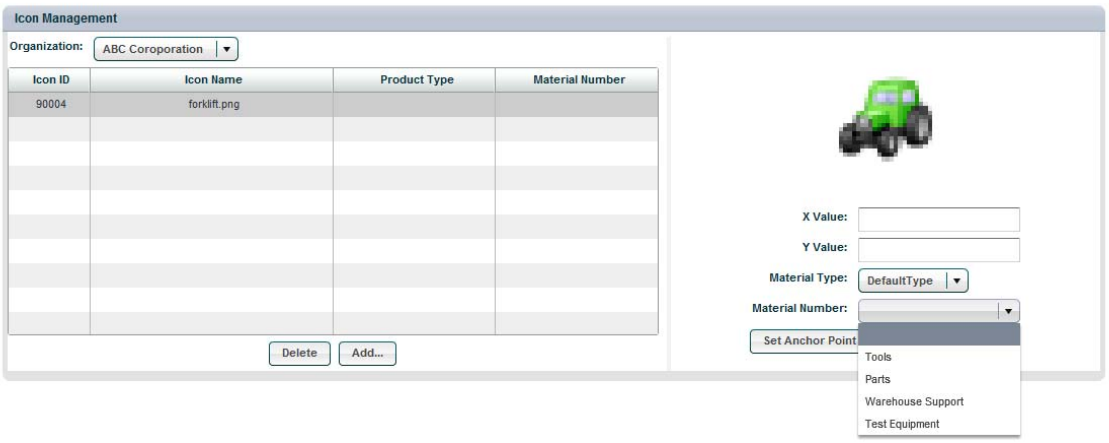
9.5 Asset Icons

Once an image is calibrated then the same can be used to show and plot the items (assets) on them to display where they are currently. While showing items (assets) on an Image using Image Maps, there is a need of showing different icons for different materials or material types. In this screen icons (usually small images) can be uploaded and associated to a material type or material.

9.5.1 Uploading an Icon:

Clicking on “Add” button pop ups a window to choose the image to upload. Choose the image to upload and click on “Open” button. This will upload the image file to server.

Now the uploaded image can be seen on the right side. Uploaded icon can be associated to a Material Type and Material. Icon can be assigned to Material Type only also. For each Material & Material Type combination at the max only one icon can be associated. Clicking on the “Save” button will save the association.



9.5.2 Setting Anchoring Point:

As any icon will have lot pixels in size, when placing the icons on a Image we need to have a point to refer as a base point for the image and this point is called the anchoring point.

When an icon is selected on the data grid table, details of the icon are displayed on the right side. To set an anchoring point of an icon, click on the button “Set Anchor Point” and move the mouse over the icon image. When the mouse is moving, the pixel values will be kept on updated in X, Y values. Now click on the icon which point has to be set an anchoring point and the same values are updated to X/Y fields. Clicking on “Save” button will save the values.

10 Flex Image Map Component

10.1 Introduction

This document describes the InSync's Image Map Component's library available in Edgeware. The components in the library consist of action scripts and additional attributes supported by the components.

The purpose of this document is to demonstrate how to use these Image Map Components while developing the user interfaces.

The main component is image map viewer and this can be used to show the markers by adding them to the component. These markers can be moved by setting their positions once again to new positions. Markers can be of type images or shapes. This component supports lot of other features also they described below in detail.

10.2 Components

The components available are

- Image Map Viewer Wrapper
- Image Map Viewer
- Image Maps Util
- Image Marker Info

10.2.1 Pre-Requisite

Step 1: The ImageMapComponent.swc should be added to the classpath of the Flex Project.

10.3 Image Map Viewer Wrapper

Usage:

Attributes: The different attributes available in the Image Map Viewer Wrapper are

Attribute	Description
currentLocation	
currentLocationKey	
offsetPixels	Default Value : 3
drawAllRegions	Default : false
childLocationRequired	Default : true
imgViewerAlign	Default : left
mobileRegionFillAlpha	Default Value : 0.3
mobileRegionFillColor	Default Color : 0xFF0000
showMobileRegions	Default : true
otherPlaneFillAlpha	Default Value : 0.2
otherPlaneFillColor	Default Color : 0xFF0000
forPrint	
imageHeight	
imageWidth	
topImageId	

Functions:

The following functions are available on the Image Map Viewer Wrapper.

- public function resizeImage():void
- public function setMarkerArray(newMarkers:ArrayCollection):void
- public function setMarker(iMarkerInfo:ImageMarkerInfo):IImageMarker
- public function removeMarker(objId:String):void
- public function removeAllMarkers():void
- public function updateRegions(regionsXML:XML):void
- public function addLinePoint(iPointInfo:ImagePointInfo):void
- public function resetImageZoom():void
- public function onBackClick(event:GenericEvent):void
- public function highlightAsset(objId:String):void
- public function resetAsset(objId:String):void
- public function resetAssets():void

- public function highlightRegion(regionName:String, highlightColor:uint, fillAlpha:Number=0.4):void

This function returns

regionName:

hightlightColor:

fillAlpha:

- public function resetRegion(regionName:String):void
- public function resetRegions():void
- public function setToolTips(toolTips:XML):void

10.4 Image Map Viewer

Usage:

Attributes: The different attributes available in the Image Map Viewer are

Attribute	Description
idlWidth	Default Value : 600
idlHeight	Default Value : 450
imageWidth	Default Value : 0
imageHeight	Default Value : 0
offsetPixels	Default Value : 3
functionToCallAfterLoad	
forPrint	Default : false
drawAllRegions	Default : false
mobileRegionFillAlpha	Default Value : 0.3
mobileRegionFillColor	Default Color : 0xFF0000
showMobileRegions	Default : true
otherPlaneFillAlpha	Default Value : 0.2
otherPlaneFillColor	Default Color : 0xFF0000
moveStep	Default Value : 0.01

Functions:

The following functions are available on the Image Map Viewer.

public function getCurrentZoom():Number

public function resetImageZoom():void

public function resizeImage():void

public function setBounds(maxZoom:Number=5, minSize:Number=200, paddingWidth:Number=75):void

This function returns

maxZoom:

minSize:

paddingWidth:

- public function resetBounds():void
- public function setOtherPlanesInfo(planesInfo:XML):void
- public function setMarker(iMarkerInfo:ImageMarkerInfo):IImageMarker
- public function addMarker(iMarkerInfo:ImageMarkerInfo):
- IImageMarker
- public function addPointToLine(iPointInfo:ImagePointInfo):void
- public function removeMarker(objId:String):void
- public function removeAllMarkers():void
- public function loadImage():void
- public function loadChildLocations(childs:XML):void
- public function loadOtherPlanes():void
- public function loadRegions(regions:XML):void
- public function updateRegions(regions:XML):void
- public function updateRegion(regionArea:RegionArea):void
- public function highlightRegion(regionName:String, highlightColor:uint, fillAlpha:Number=0.4):void

This function returns

regionName:

highlightColor:

fillAlpha:

- public function resetRegion(regionName:String):void
- public function resetRegions():void
- public function highlightAsset(objId:String):void
- public function resetAsset(objId:String):void
- public function resetAssets():void
- public function setToolTips(toolTips:XML):void
- public function isNeededForPrint(regName:String):void
- public function setMarkerFilters(newFilters:ArrayCollection):void
- public function removeMarkerFilters():void
- public function getCoordinateValuesForLocalPoint(localPoint:Point):Point

10.5 Image Maps Util

Usage:

Functions:

The following functions are available on the Image Maps Util.

public static function getImagePoint(calibrationInfo:ImageCalibrationInfo, coordinate1Value:Number, coordinate2Value:Number):Point

This function returns

calibrationInfo:

coordinate1Value:

coordinate2Value:

public static function getCoordinateValuesForImagePoint (calibrationInfo:ImageCalibrationInfo, imagePoint:Point):Point

This function returns

calibrationInfo:

imagePoint:Image Marker Info

Usage:Attributes: The different attributes available in the Image Marker Info are

Attribute	Description
objId	
objType	
objDesc	
distanceX	
distanceY	
distanceZ	
shape	
shapeValues	
shapeFillColor	Default Color : 0xFF00FF
shapeFillAlpha	Default Value : 1
shapeBorderRequired	Default : false
imageId	
imageURL	
anchorX	Default Value : 0
anchorY	Default Value : 0
markerBorderColor	Default Color : 0x0000FF
markerBorderThickness	Default Value : 1

markerText	
checkForExistence	Default : true
offsetRequired	Default : false
moveAnimationTime	Default Value : 2000
isAnimationRequiredToMove	Default : true
markerUpdationRequired	Default : false
accuracy	
accuracyCircleRadius	Default : 0
accuracyCircleFillColor	Default Color : 0x0000FF
accuracyCircleFillAplpha	Default Value : 0.3

11 Custom DataGrid Capabilities

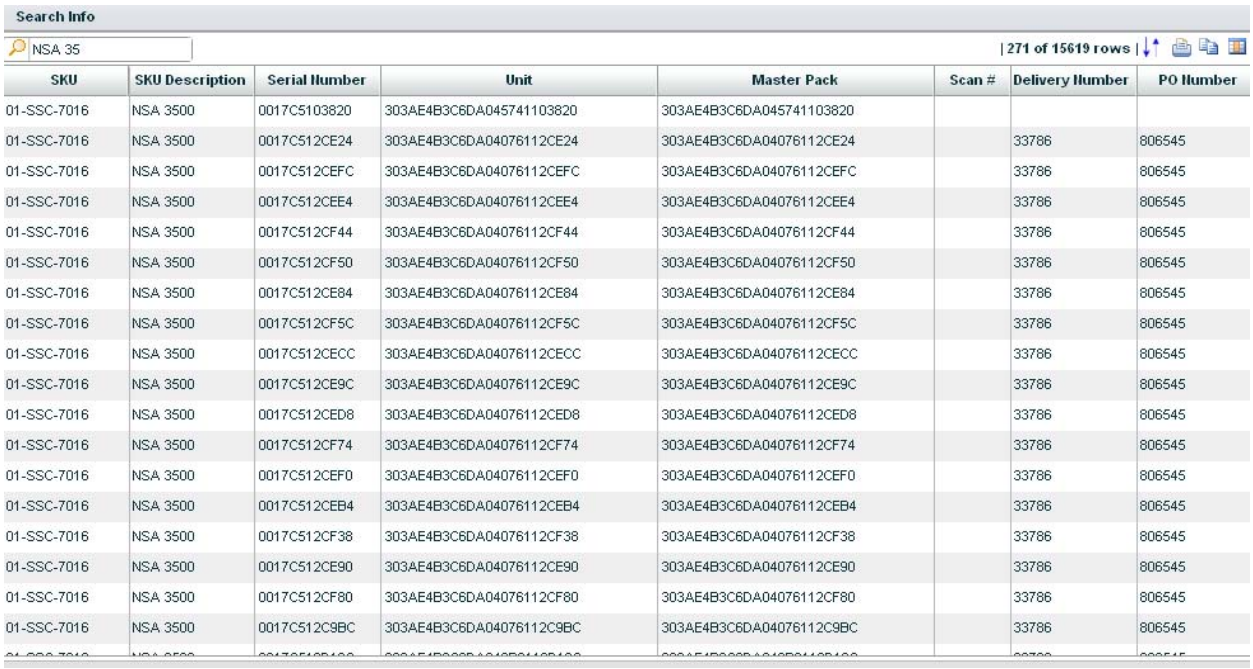
11.1 Custom DataGrid

InSync's Custom Datagrid has the following capabilities.

- Context Search
- Column Filters
- Multi Column Sort
- Print Columns
- Clipboard Copy
- Column Selector for the view

11.1.1 Context Search:

Key in the text being searched for into the Search text box. The grid filters automatically as the text is getting keyed in. The Context search is against all the columns which are visible in the grid. The text being searched for can be from any of the columns. For ex. keying in NSA 3500 in the text box brings up the following results.


Search Info

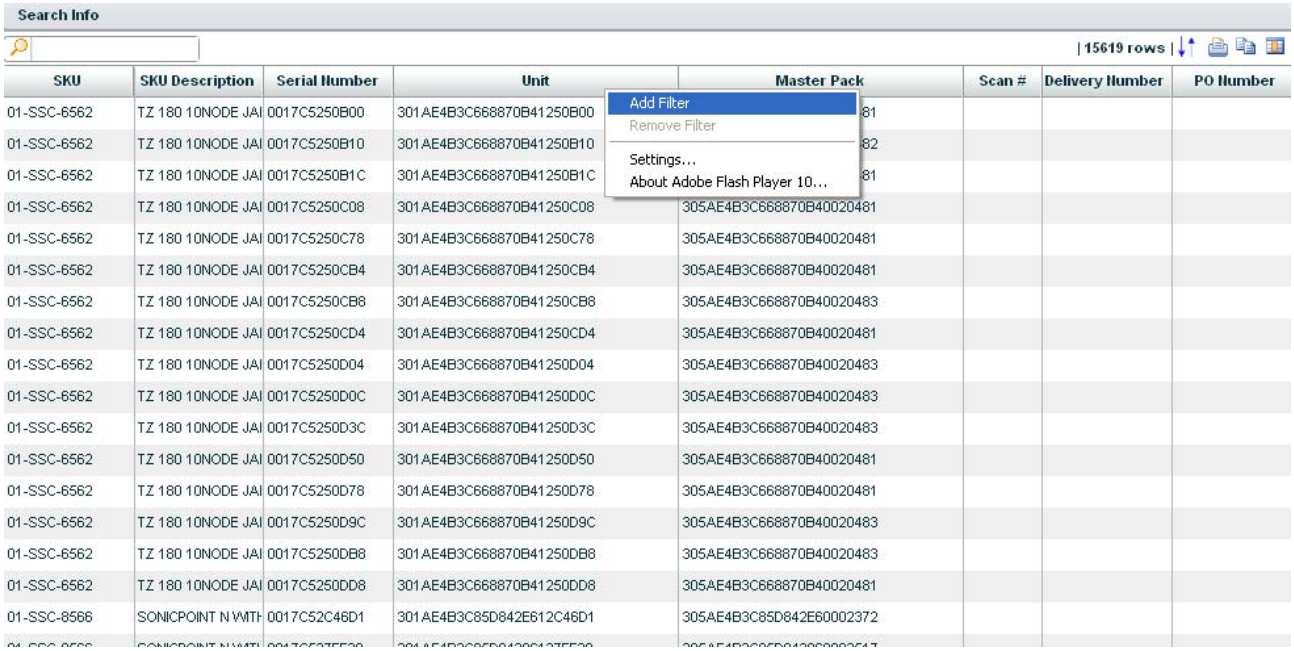
NSA 35

| 271 of 15619 rows |

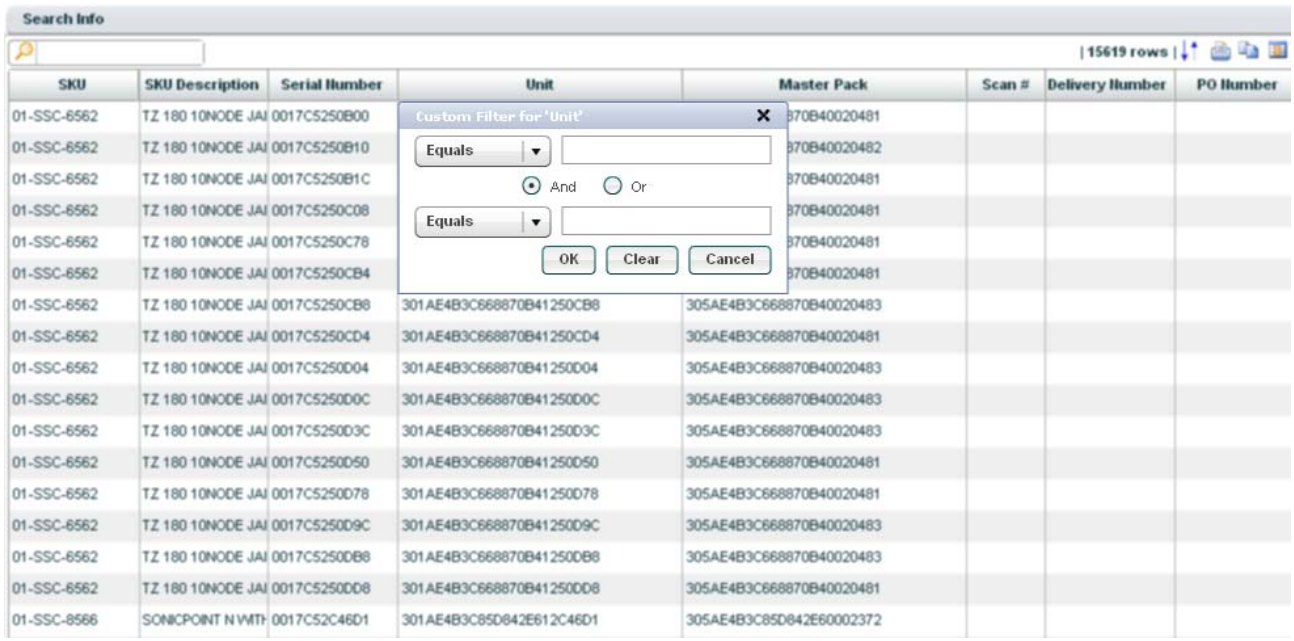
SKU	SKU Description	Serial Number	Unit	Master Pack	Scan #	Delivery Number	PO Number
01-SSC-7016	NSA 3500	0017C5103820	303AE4B3C6DA045741103820	303AE4B3C6DA045741103820			
01-SSC-7016	NSA 3500	0017C512CE24	303AE4B3C6DA04076112CE24	303AE4B3C6DA04076112CE24		33786	806545
01-SSC-7016	NSA 3500	0017C512CEFC	303AE4B3C6DA04076112CEFC	303AE4B3C6DA04076112CEFC		33786	806545
01-SSC-7016	NSA 3500	0017C512CEE4	303AE4B3C6DA04076112CEE4	303AE4B3C6DA04076112CEE4		33786	806545
01-SSC-7016	NSA 3500	0017C512CF44	303AE4B3C6DA04076112CF44	303AE4B3C6DA04076112CF44		33786	806545
01-SSC-7016	NSA 3500	0017C512CF50	303AE4B3C6DA04076112CF50	303AE4B3C6DA04076112CF50		33786	806545
01-SSC-7016	NSA 3500	0017C512CE84	303AE4B3C6DA04076112CE84	303AE4B3C6DA04076112CE84		33786	806545
01-SSC-7016	NSA 3500	0017C512CF5C	303AE4B3C6DA04076112CF5C	303AE4B3C6DA04076112CF5C		33786	806545
01-SSC-7016	NSA 3500	0017C512CECC	303AE4B3C6DA04076112CECC	303AE4B3C6DA04076112CECC		33786	806545
01-SSC-7016	NSA 3500	0017C512CE9C	303AE4B3C6DA04076112CE9C	303AE4B3C6DA04076112CE9C		33786	806545
01-SSC-7016	NSA 3500	0017C512CED8	303AE4B3C6DA04076112CED8	303AE4B3C6DA04076112CED8		33786	806545
01-SSC-7016	NSA 3500	0017C512CF74	303AE4B3C6DA04076112CF74	303AE4B3C6DA04076112CF74		33786	806545
01-SSC-7016	NSA 3500	0017C512CEF0	303AE4B3C6DA04076112CEF0	303AE4B3C6DA04076112CEF0		33786	806545
01-SSC-7016	NSA 3500	0017C512CEB4	303AE4B3C6DA04076112CEB4	303AE4B3C6DA04076112CEB4		33786	806545
01-SSC-7016	NSA 3500	0017C512CF38	303AE4B3C6DA04076112CF38	303AE4B3C6DA04076112CF38		33786	806545
01-SSC-7016	NSA 3500	0017C512CE90	303AE4B3C6DA04076112CE90	303AE4B3C6DA04076112CE90		33786	806545
01-SSC-7016	NSA 3500	0017C512CF80	303AE4B3C6DA04076112CF80	303AE4B3C6DA04076112CF80		33786	806545
01-SSC-7016	NSA 3500	0017C512C9BC	303AE4B3C6DA04076112C9BC	303AE4B3C6DA04076112C9BC		33786	806545

11.1.2 Column Filters

Right Click on the column where the filter needs to be added.

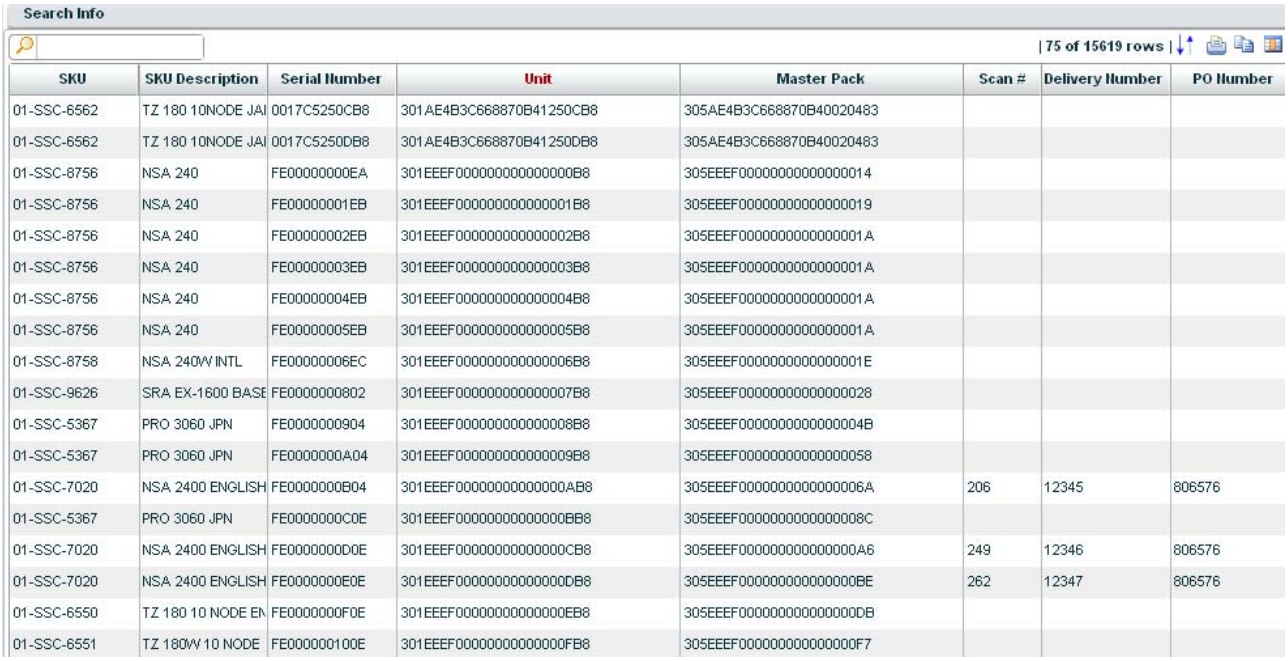


Click ‘Add Filter’



Select the filtering criteria from the drop down and key in the text in the text box. Select ‘And’, ‘Or’ buttons based on the need.

For ex – Selecting ‘Ends With’ from the drop down and with text as ‘B8’ the grid results are as shown below.

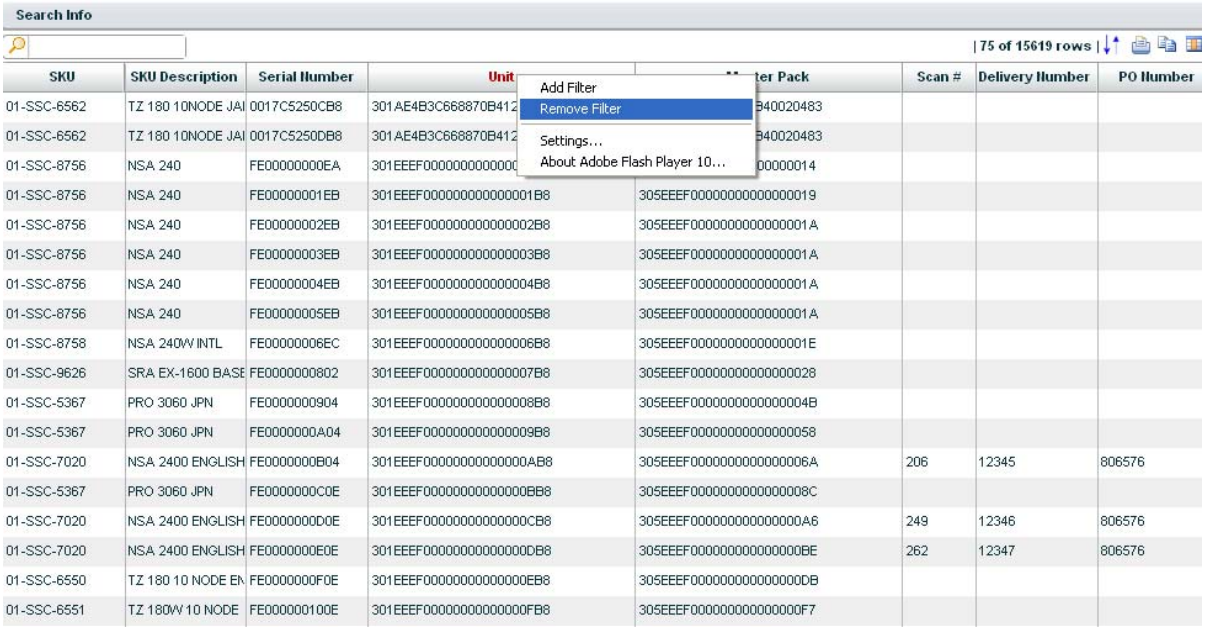

Search Info

| 75 of 15619 rows |

SKU	SKU Description	Serial Number	Unit	Master Pack	Scan #	Delivery Number	PO Number
01-SSC-6562	TZ 180 10NODE JAI	0017C5250CB8	301AE4B3C668870B41250CB8	305AE4B3C668870B40020483			
01-SSC-6562	TZ 180 10NODE JAI	0017C5250DB8	301AE4B3C668870B41250DB8	305AE4B3C668870B40020483			
01-SSC-8756	NSA 240	FE00000000EA	301EEEF000000000000000B8	305EEEF00000000000000014			
01-SSC-8756	NSA 240	FE00000001EB	301EEEF000000000000001B8	305EEEF00000000000000019			
01-SSC-8756	NSA 240	FE00000002EB	301EEEF000000000000002B8	305EEEF0000000000000001A			
01-SSC-8756	NSA 240	FE00000003EB	301EEEF000000000000003B8	305EEEF0000000000000001A			
01-SSC-8756	NSA 240	FE00000004EB	301EEEF000000000000004B8	305EEEF0000000000000001A			
01-SSC-8756	NSA 240	FE00000005EB	301EEEF000000000000005B8	305EEEF0000000000000001A			
01-SSC-8758	NSA 240W INTL	FE00000006EC	301EEEF000000000000006B8	305EEEF0000000000000001E			
01-SSC-9626	SRA EX-1600 BASE	FE0000000802	301EEEF000000000000007B8	305EEEF00000000000000028			
01-SSC-5367	PRO 3060 JPN	FE0000000904	301EEEF000000000000008B8	305EEEF0000000000000004B			
01-SSC-5367	PRO 3060 JPN	FE0000000A04	301EEEF000000000000009B8	305EEEF00000000000000058			
01-SSC-7020	NSA 2400 ENGLISH	FE0000000B04	301EEEF00000000000000AB8	305EEEF0000000000000006A	206	12345	806576
01-SSC-5367	PRO 3060 JPN	FE0000000C0E	301EEEF00000000000000BB8	305EEEF0000000000000008C			
01-SSC-7020	NSA 2400 ENGLISH	FE0000000D0E	301EEEF00000000000000CB8	305EEEF000000000000000A6	249	12346	806576
01-SSC-7020	NSA 2400 ENGLISH	FE0000000E0E	301EEEF00000000000000DB8	305EEEF000000000000000BE	262	12347	806576
01-SSC-6550	TZ 180 10 NODE EN	FE0000000F0E	301EEEF00000000000000EB8	305EEEF000000000000000DB			
01-SSC-6551	TZ 180W 10 NODE	FE000000100E	301EEEF00000000000000FB8	305EEEF000000000000000F7			

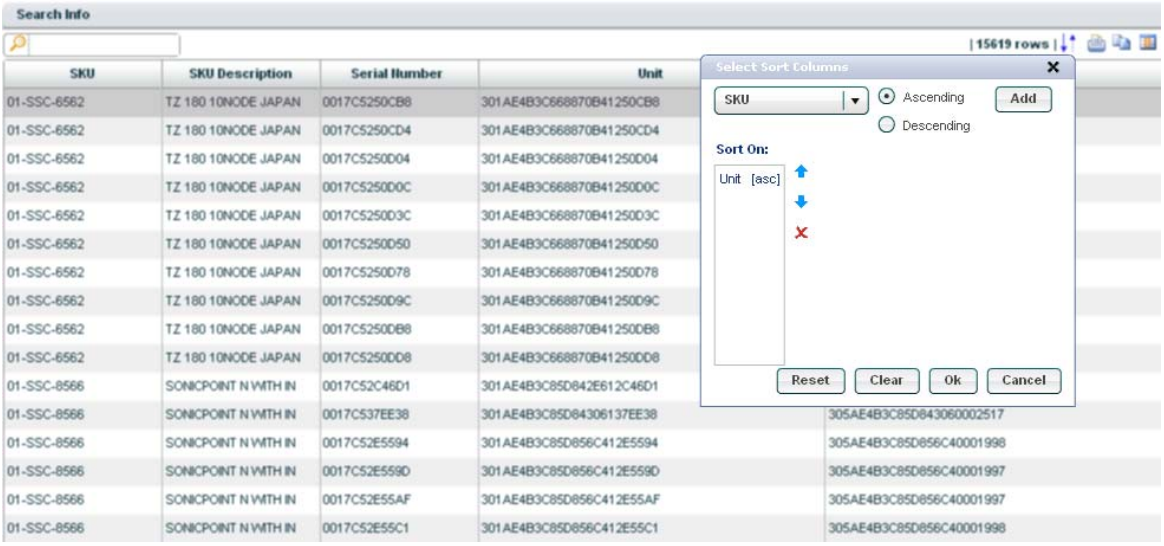
The columns where the filters have been added are marked in red for identification purpose. Filters can be added on multiple columns at the same time.

To remove the filters right click on the column where there is a filter and click ‘Remove Filter’. This will remove the filter.



11.1.3 Multi Column Sort

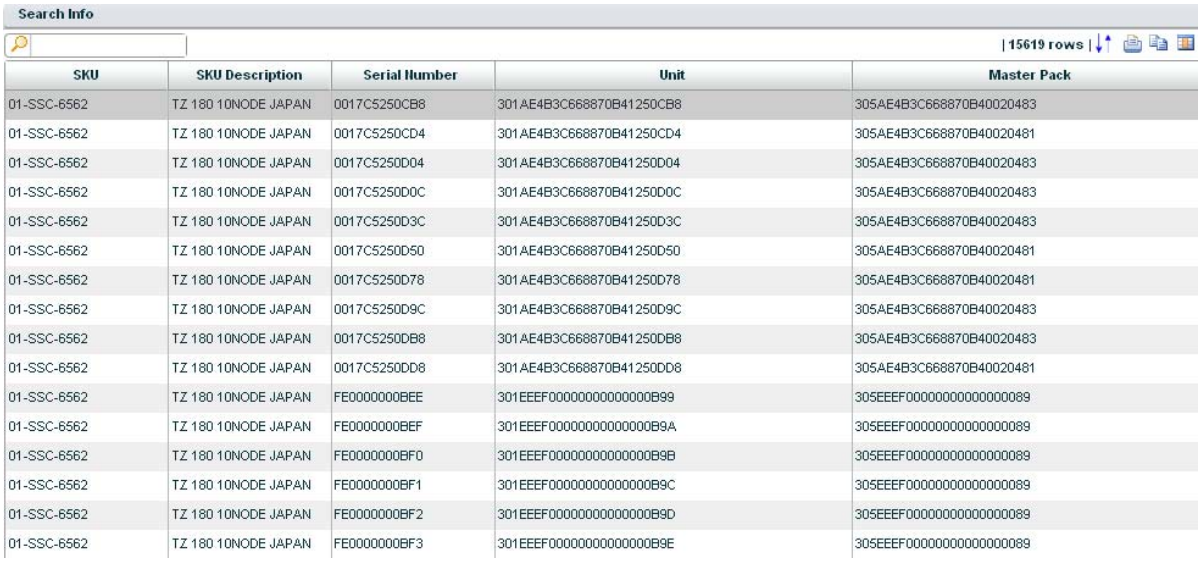
Sort can be achieved on a single / multiple columns by clicking on the icon located on the top right side corner of the grid.



Select the column from the drop down, select the ascending/descending radio button and click ‘Add’. This will add the column to the Sort On list box. Follow the same procedure to add multiple columns to the List box. The sort order can be modified using the ⬆ or the ⬇ buttons. To delete the column from the Sort list, select the column and click on ✖ button. Click ‘Ok’ to set the Sort parameters and view the results on the grid.

Click ‘Clear’ to clear the sort selection. Click on ‘Reset’ to reset to the previous selection on any change to the Sort selection.

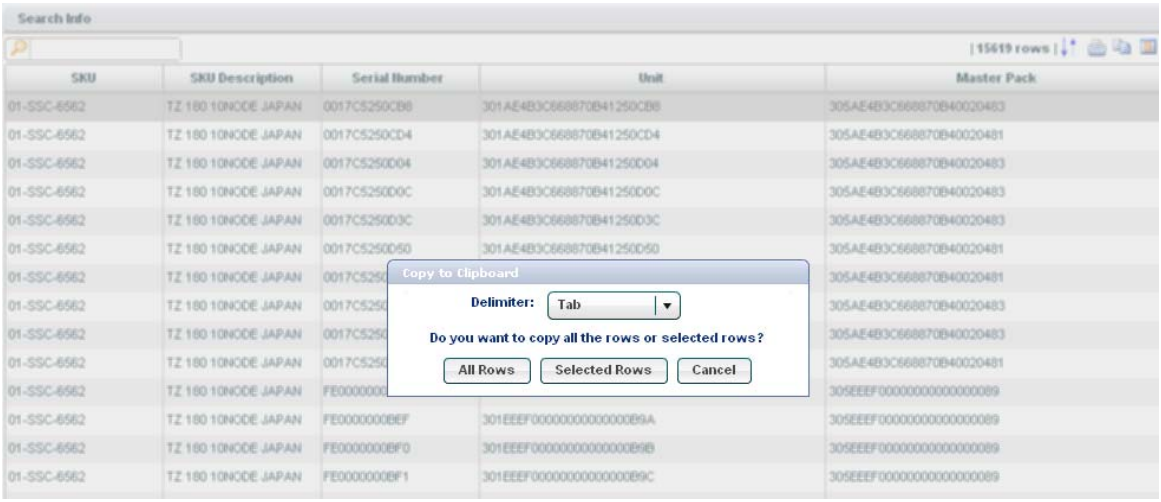
For ex. Selecting SKU (desc) and Unit (asc) as the sort columns the grid results are as shown below.



SKU	SKU Description	Serial Number	Unit	Master Pack
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250CB8	301AE4B3C668870B41250CB8	305AE4B3C668870B40020483
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250CD4	301AE4B3C668870B41250CD4	305AE4B3C668870B40020481
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250D04	301AE4B3C668870B41250D04	305AE4B3C668870B40020483
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250D0C	301AE4B3C668870B41250D0C	305AE4B3C668870B40020483
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250D3C	301AE4B3C668870B41250D3C	305AE4B3C668870B40020483
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250D50	301AE4B3C668870B41250D50	305AE4B3C668870B40020481
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250D78	301AE4B3C668870B41250D78	305AE4B3C668870B40020481
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250D9C	301AE4B3C668870B41250D9C	305AE4B3C668870B40020483
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250DB8	301AE4B3C668870B41250DB8	305AE4B3C668870B40020483
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250DD8	301AE4B3C668870B41250DD8	305AE4B3C668870B40020481
01-SSC-6562	TZ 180 10NODE JAPAN	FE0000000BEE	301EEEF00000000000000B99	305EEEF00000000000000089
01-SSC-6562	TZ 180 10NODE JAPAN	FE0000000BEF	301EEEF00000000000000B9A	305EEEF00000000000000089
01-SSC-6562	TZ 180 10NODE JAPAN	FE0000000BF0	301EEEF00000000000000B9B	305EEEF00000000000000089
01-SSC-6562	TZ 180 10NODE JAPAN	FE0000000BF1	301EEEF00000000000000B9C	305EEEF00000000000000089
01-SSC-6562	TZ 180 10NODE JAPAN	FE0000000BF2	301EEEF00000000000000B9D	305EEEF00000000000000089
01-SSC-6562	TZ 180 10NODE JAPAN	FE0000000BF3	301EEEF00000000000000B9E	305EEEF00000000000000089

11.1.4 Clipboard Copy

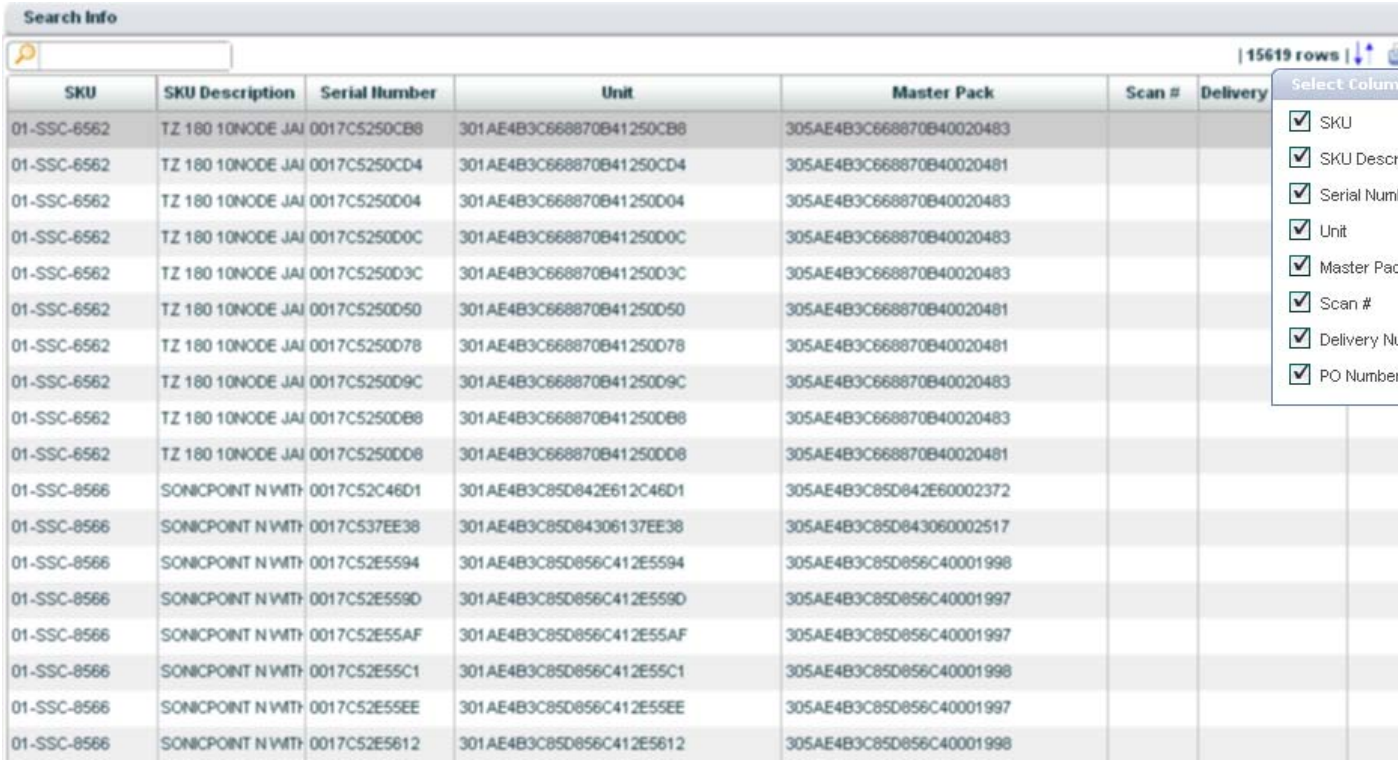
All the rows or a selected row can be copied to the clipboard using the clipboard copy feature. This is possible by clicking on located on the top right side corner of the grid.



Select the Delimiter from the drop down and click on ‘All Rows’ or ‘Selected Row’ button. The results can be pasted to any text editor.

11.1.5 Column Selector

The columns visible on the data grid can be selected/de-selected based on the view required. This is possible by clicking the icon located on the top right side corner of the grid.



Select/deselect the check boxes against the columns for the required view of the grid. For ex deselect the Scan #, Delivery Number and the PO Number

Search Info

| 15619 rows |

SKU	SKU Description	Serial Number	Unit	Master Pack
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250CB8	301AE4B3C668870B41250CB8	305AE4B3C668870B40020483
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250CD4	301AE4B3C668870B41250CD4	305AE4B3C668870B40020481
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250D04	301AE4B3C668870B41250D04	305AE4B3C668870B40020483
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250D0C	301AE4B3C668870B41250D0C	305AE4B3C668870B40020483
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250D3C	301AE4B3C668870B41250D3C	305AE4B3C668870B40020483
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250D50	301AE4B3C668870B41250D50	305AE4B3C668870B40020481
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250D78	301AE4B3C668870B41250D78	305AE4B3C668870B40020481
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250D9C	301AE4B3C668870B41250D9C	305AE4B3C668870B40020483
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250DB8	301AE4B3C668870B41250DB8	305AE4B3C668870B40020483
01-SSC-6562	TZ 180 10NODE JAPAN	0017C5250DD8	301AE4B3C668870B41250DD8	305AE4B3C668870B40020481
01-SSC-8566	SONICPOINT N WITH IN	0017C52C46D1	301AE4B3C85D842E612C46D1	305AE4B3C85D842E60002372
01-SSC-8566	SONICPOINT N WITH IN	0017C537EE38	301AE4B3C85D84306137EE38	305AE4B3C85D843060002517
01-SSC-8566	SONICPOINT N WITH IN	0017C52E5594	301AE4B3C85D856C412E5594	305AE4B3C85D856C40001998
01-SSC-8566	SONICPOINT N WITH IN	0017C52E559D	301AE4B3C85D856C412E559D	305AE4B3C85D856C40001997
01-SSC-8566	SONICPOINT N WITH IN	0017C52E55AF	301AE4B3C85D856C412E55AF	305AE4B3C85D856C40001997
01-SSC-8566	SONICPOINT N WITH IN	0017C52E55C1	301AE4B3C85D856C412E55C1	305AE4B3C85D856C40001998

12 Load Balancing

12.1 Introduction

This document describes the steps involved in setting up the load balancer using Apache Web server. The version used for the purpose is 2.2.13 and uses the mod_proxy module.

The load balancer setup explained in the document uses 3 servers – one load balancer (Apache Web server), and 2 sensornet servers.

For the purpose of documentation we will assume that the load balancer is running on server 'iApp-lb' (default port 80) and the 2 sensornet instances running on iApp-sn1 and iApp-sn2 servers respectively (on ports 8080).

Note: Pls refer to the vendor's documentation in setting up the load balancer with Tomcat if the load balancer software is different from Apache Server

12.2 Minimum Hardware and Software Requirements

a. Windows XP Professional for POC and Development environments / Windows Server 2003 for production use.

b. Core 2 Duo / Quad Core Processor

c. 3 GB RAM

d. 60 GB SAS Hard Disk

12.3 Load Balancer - Pre-Requisites

1. Install the 2 instances of Sensornet. Refer to IApp Installation section to install Sensornet. Make sure that both the instances point to the same database.

2. Login to http://iApp-sn1:8080/sensornet and http://iApp-sn2:8080/sensornet to check if the setup is configured properly.

3. Login to http://iApp-sn1:8080/sensornet. Navigate to Administration->Instance Registry and setup 2 registry entries for sensornet. Note that for both the instances the MODE is set to 'PRIMARY'. Refer to Instance Registry section for help on setting up the Instance Registry.

Note – Step 3 is an important configuration step to be setup for load balancing to work.

4. Download Apache and install on iApp-lb server as a Windows service. Refer to Apache installation manual for help during the installation.

5. Open the browser and open http://iApp-lb. You should see the default Apache 'It Works' page in the web page if the installation is proper.

12.4 Configure Apache for Load Balancing

1. Open httpd.conf file located in <Apache_install_folder>/conf in a file editor.

2. Enable the following modules in the LoadModules section.

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
LoadModule proxy_balancer_module modules/mod_proxy_balancer.so
LoadModule proxy_connect_module modules/mod_proxy_connect.so
LoadModule proxy_http_module modules/mod_proxy_http.so
LoadModule rewrite_module modules/mod_rewrite.so
LoadModule status_module modules/mod_status.so
```

Note: Refer to Apache website download section for the missing modules in your installation. Ideally all the modules should be available in the base apache installation

3. Add the following lines at the end of the file and Save the file

```
**********************************************************************************************
ProxyPass /sensornet balancer://sensor/sensornet stickysession=JSESSIONID nocanon
ProxyPassReverse /sensornet balancer://sensor/sensornet

 <Proxy balancer://sensor>
          BalancerMember http://iApp-sn1:8080 route=sn1 loadfactor=50 min=0 max=10
          BalancerMember http://iApp-sn2:8080 route=sn2 loadfactor=50 min=0 max=10

 </Proxy>

<Location /balancer-manager>
         SetHandler balancer-manager
</Location>
**********************************************************************************************
```

4. On the server iApp-sn1 open the server.xml located in the <tomcat_install_dir>/conf folder. Add jvmRoute attribute to the Engine element.

```
  <Engine name="catalina" defaultHost="localhost" jvmRoute="sn1">
```

The value ‘sn1’ is the value configured to the root attribute of the BalancerMember for iApp-sn1.

5. On the server iApp-sn2 open the server.xml located in the <tomcat_install_dir>/conf folder. Add jvmRoute attribute to the Engine element.

```
  <Engine name="catalina" defaultHost="localhost" jvmRoute="sn2">
```

The value ‘sn2’ is the value configured to the root attribute of the BalancerMember for iApp-sn2.

6. Restart both the tomcat servers.

7. Restart Apache.

8. To Access the application – Open the browser and type in http://iApp-lb/sensornet/

13 iApp Administration Manual

13.1 Introduction

This document is meant for administrators of RFID IApp™. This is the starting point for setting up IApp.

13.2 Administration Set up

Administration section allows administrators to set up the Enterprise Organization structure which is the key to Configuration in IApp. The whole user management for the enterprise organization can be setup using the Administration feature. Nothing can be configured in IApp without creating the Enterprise organization.

Notes:

- Home/Managed Enterprise is created at the time of IApp installation.
- Other Enterprise Organizations are created within the Home Enterprise.
- Only the administrator at the Home Enterprise will be able to administer all the other Enterprises.
- Users of a particular Enterprise will have access to that Enterprise only except users at the Home Enterprise who can be gives privileges to other Enterprise Organizations.

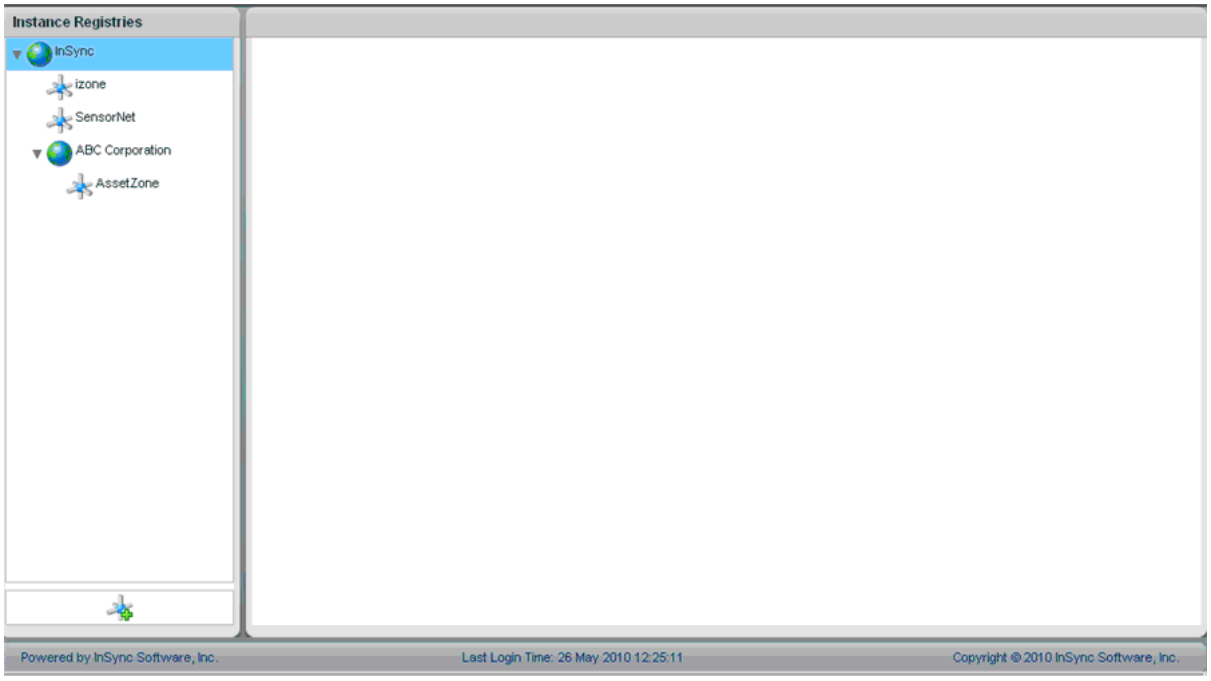
13.3 Activities

Following are the administrative activities (in the order of the priority).

- Instance Registry
- Organizations / Users
- Roles
- Catalogs
- Performance Tuning
- Messages
- User Locks
- Organization Locks

13.4 Instance Registry

The instance registry is the starting point in setting up IApp Sensornet and Process Zones after the installation. The different instances of Sensornet and the Process Zones are setup here before running the application.



Note:

1. Sensornet instances are always created at the Home Enterprise

2. Each Enterprise will have the zone instances created.

The following is setup using the instance registry

- Sensornet details at the Home Enterprise
- Process Zones instances at the respective enterprise organization.
- Configure primary and backup Sensornet for hot stand by option.

Note:

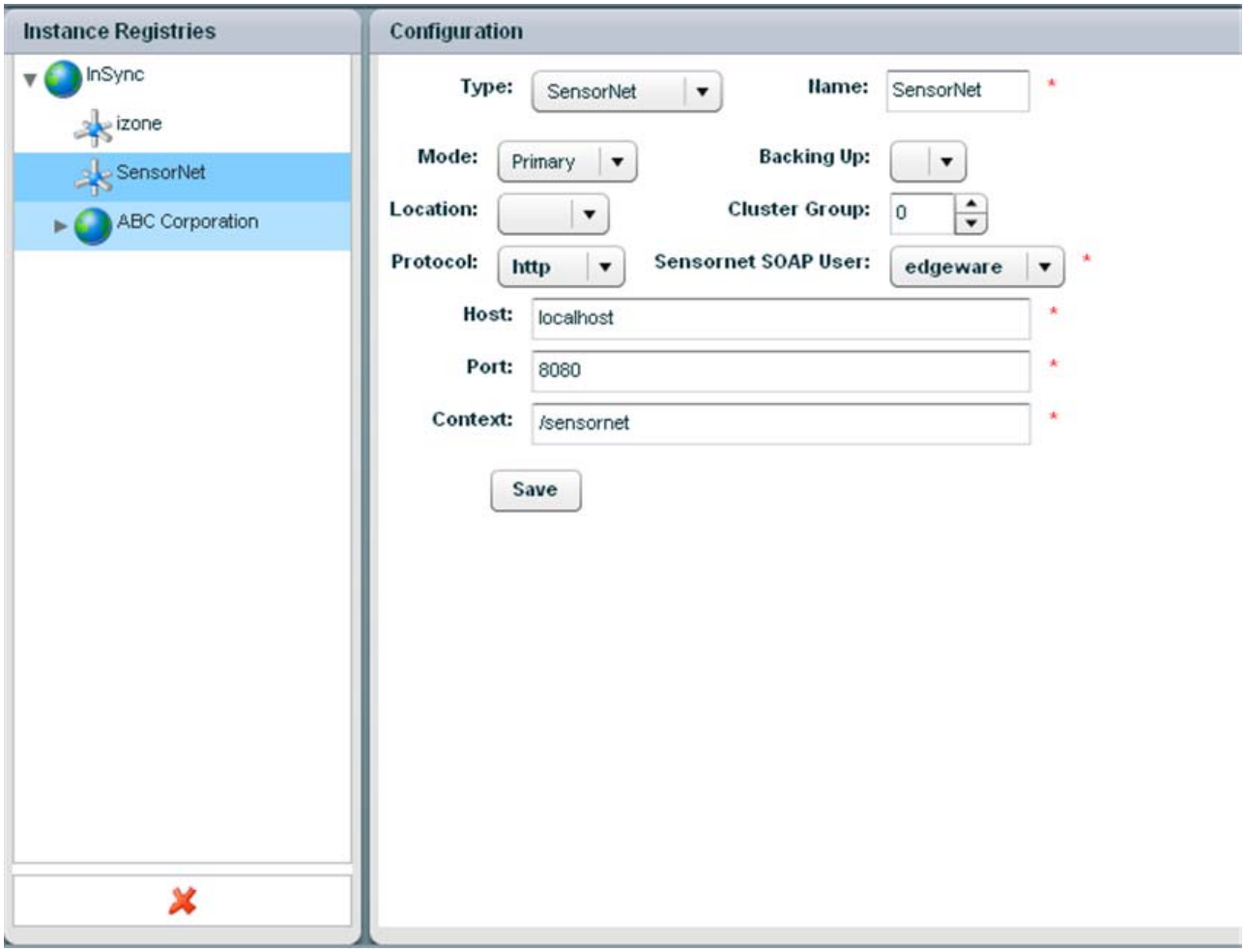
1. It is important that Sensornet, Process Zones and the Print Servers installed using the installer are configured in the instance registry for the communication to happen between them.

2. Hot stand by feature is not supported in this release.

13.5 Create Sensornet Instance

- Select the Home Enterprise in the Instance Pools panel and click

- Select ‘SensorNet’ from the Type drop down.



Note: IApp provides a sesornet instance configured as part of the installation. Review and make changes according to your setup.

The different attributes are

Attribute	Description
Type	Type of instance to create. ‘Sensornet’ in this case
Name	Name of the Sensornet. This will be the instance name given while creating the Sensornet using the installer. **Note:** The same name should be configured as the instance name in the config.xml of sensornet for the application service element.
Mode	Mode of the instance. Primarily tells whether the Sensornet instance getting created is a Primary or a back up instance. If it is a ‘Backup’ mode capture the Backing Up

	Sensornet instance to which it is a backup. **Note:** If a load balancer is setup then the second sensornet instance also should be setup as a primary instance
Backing Up	If the Mode selected is BackUp then select the back up sensornet instance from the drop down.
Location	Location at which the instance is installed. Note that this attribute will get added after the location configuration is setup from the Configuration section.
Cluster Group	Logical group of instances created. The setup can have multiple cluster groups with instances set up at each cluster group. This is helpful for Central Monitoring where in Sensornet instances from different cluster groups can be monitored from a single place.
Protocol	Mode of communication. Supports 'http' / 'https' **Note:** If https protocol is selected then necessary configuration should be made in the server.xml of tomcat to support the protocol. The https connector should be enabled in the server.xml. Refer to Enabling SSL in IApp section for information on setting up SSL.
Sensornet SOAP User	User of type SYSTEM created in IApp will be listed here. This SOAP user is used while communicating between 2 sensornet instances setup on a load balancer or on a backup mode.
Host	IP address / System name where the instance has been deployed
Port	Web Server Port on which the instance is deployed and running. Pls check with the administrator on the ports to be used for http and https protocol.
Context	This is the tomcat context setup in the server.xml. Note that the context should match to the configuration in server.xml

- Click Save to save the instance to the tree.
- Select the instance from the tree to edit the instance created.

Note:

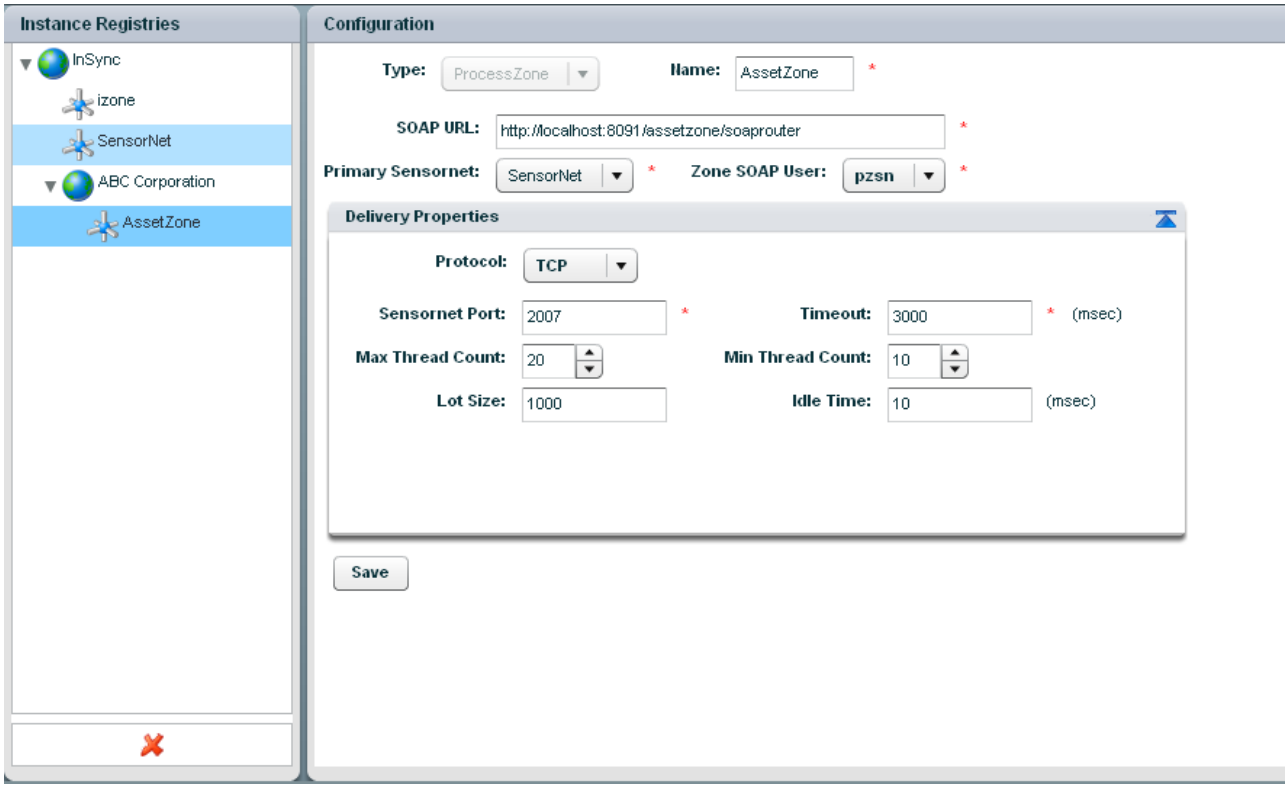
1. The protocol details are used by the Process Zone to communicate to the Sensornet for the configuration (Location,Devices,Edge Processes…) information.

2. If there are changes in the sensornet instance details then the same changes have to made in the config.xml (soapURL of the InitialSensorNetDiscovery element) of the process zone setup to communicate with the sensornet instance.

3. The SOAP user is used for authentication between two sensornet instances.

13.6 Create Process Zone Instance

- Select the Enterprise Organization in which the Zone instance has to be created and click .
- Select ProcessZone from the Type drop down.



The different attributes are

Attribute	Description
Type	Type of instance to create. ''ProcessZone' in this case.
Name	Name of the Process Zone instance. This is the name given to the zone at the time of its installation.
Primary Sensornet	The primary Sensornet instance to which the zone will communicate. Select the Sensornet

	instance from the drop down.
Zone SOAP User	All users of type PZ_SN are listed in the drop down. This is the user to communicate from process zone to sensornet. **Note:** The same user and the respective password will have to be given in the config.xml of the process zone provided in the config.xml (username, password of InitialSensorNetDiscovery element).
SOAP URL	Change the port and the zone in the URL. Where port is the port of the tomcat instance hosting the process zone the zone is the tomcat context for the process zone mentioned in the server.xml of the zone tomcat instance

Delivery Properties

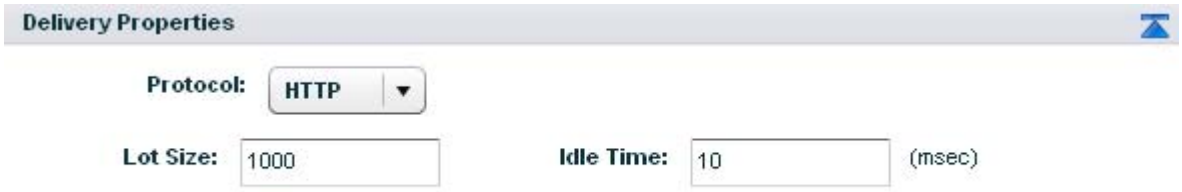
Defines the delivery mechanism between the Process Zone and Sensornet. This is basically the mode of communication between the zones and the Sensornet instances to deliver the events.

Supported protocols are TCP / HTTP / HTTPS

TCP

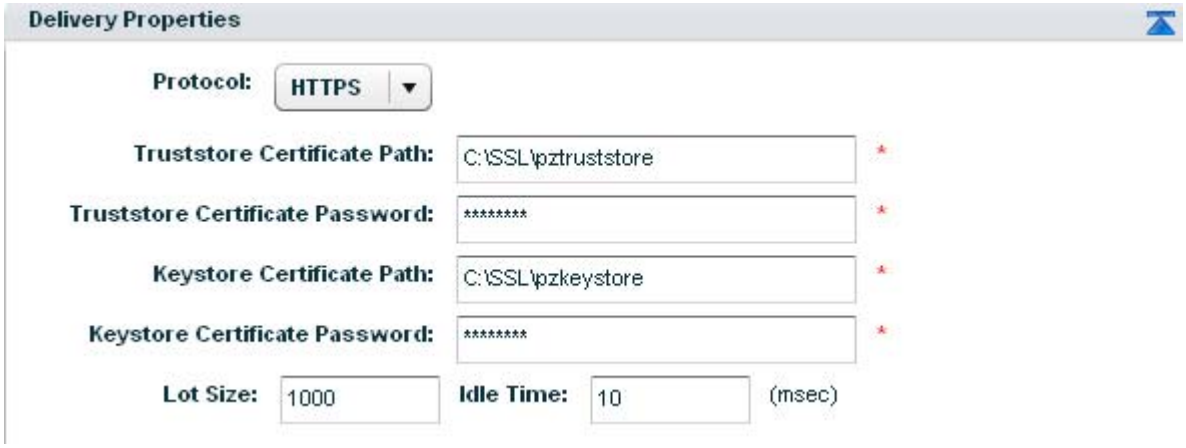
Sensornet Port	is the tcp port to communicate
Timeout	Time in millisec after which the connection will timeout while trying to establish the connection
Max Thread Count	is the max no of threads sensornet will spawn to process data from process zone.
Min Thread Count	is the minimum no of threads sensornet will spawn to process data from process zone
Lot Size	is the no of events that gets delivered to sensornet in one delivery cycle
Idle Time	is the sleep time after each event delivery cycle.

HTTP



Lot Size	is the no of events that gets delivered to sensornet in one delivery cycle
Idle Time	is the sleep time after each event delivery cycle.

HTTPS

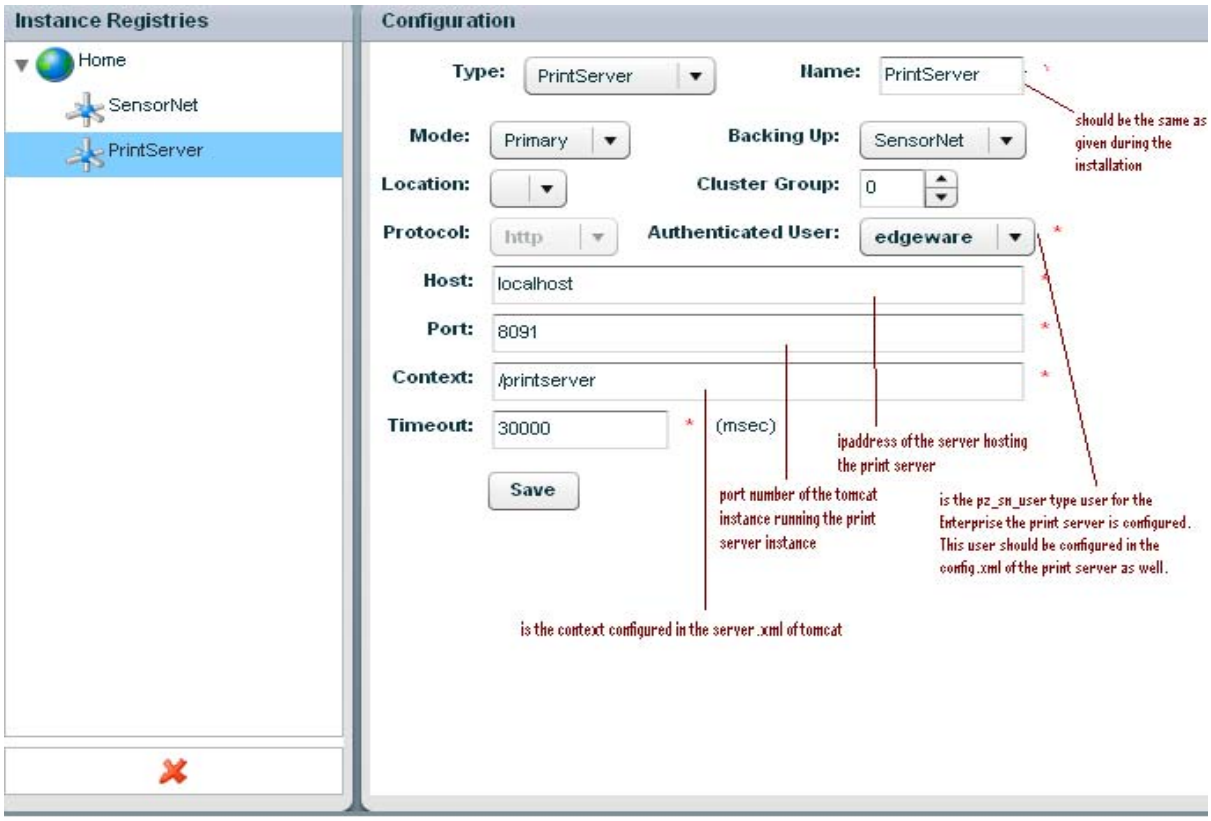


Truststore Certificate Path	Path of the certificate on the file system
Truststore Certificate password	Password for the truststore
Keystore Certificate Path	Path of the certificate on the file system
Keystore Certificate Password	Password for the keystore
Lot Size	is the no of events that gets delivered to sensornet in one delivery cycle
Idle Time	is the sleep time after each event delivery cycle.

- Click Save to save the instance to the tree.
- Select the instance from the tree to edit the instance created.

13.7 Create Print Server Instance

- Select the Enterprise Organization for which the print server instance has to be created and click .
- Select PrintServer from the Type drop down.



The different attributes are

Attribute	Description
Type	Type of instance to create. ‘’PrintServer’ in this case.
Name	Name of the Print Server instance. This is the name given to the print server at the time of its installation.
Mode	*For future use*
Backing Up	*For future use*
Location	*For future use*
Cluster Group	*For future use*
Protocol	Mode of communication. Supports ‘http’ in this version
Authenticated User	All users of type PZ_SN are listed in the drop down.

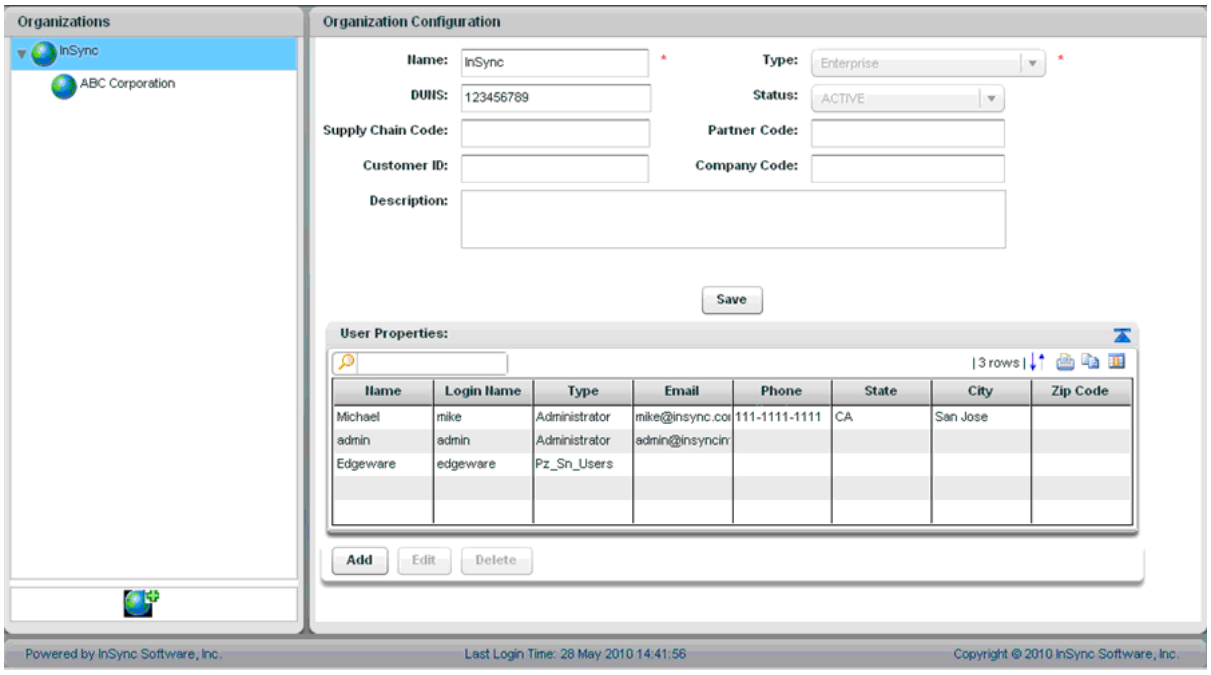
	This is the user to communicate from print server to sensornet. **Note:** The same user and the respective password will have to be given in the config.xml of the print server (username, password of Service element).
Host	IP address / System name where the instance has been deployed
Port	Web Server Port on which the instance is deployed and running. Pls check with the administrator on the ports to be used for http protocol.
Context	This is the tomcat context setup in the server.xml. Note that the context should match to the configuration in server.xml
Timeout	Time in millisec after which the connection will timeout while trying to establish the connection

- Click Save to save the instance to the tree.
- Select the instance from the tree to edit the instance created.

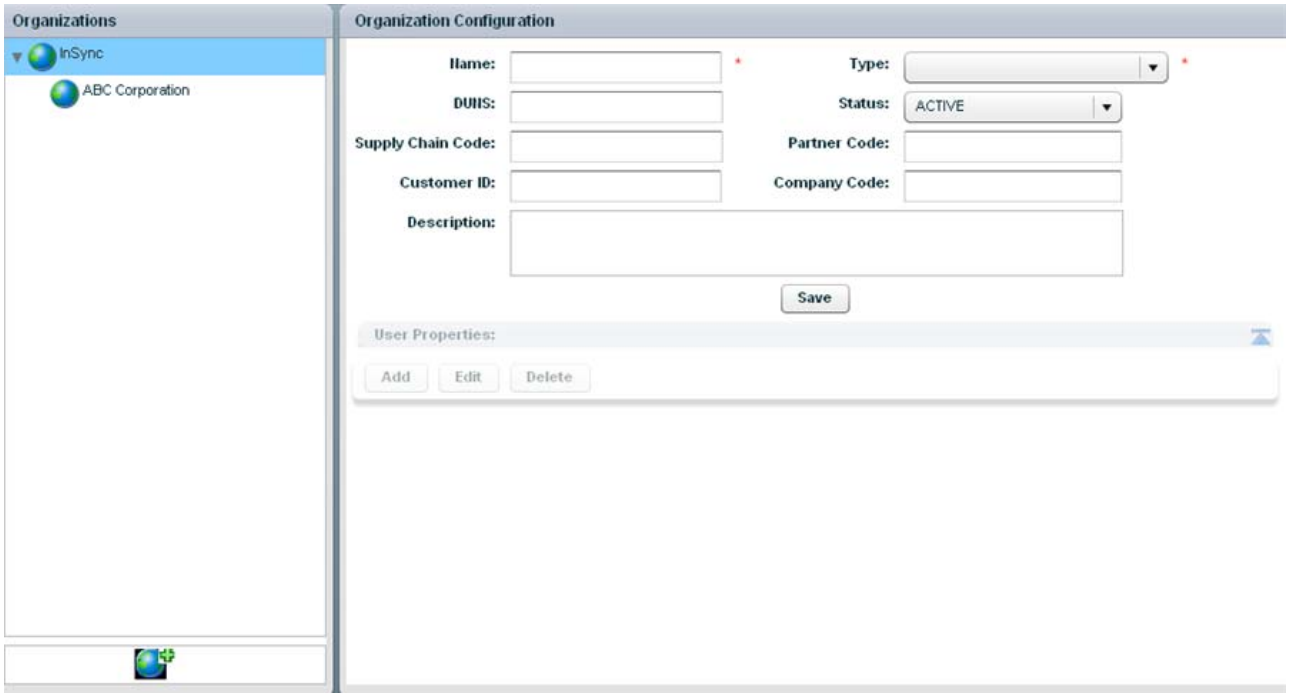
Note: Multiple print server instances can be configured for each enterprise organization

13.8 Organization / Users

Enterprise Organization is the key to the entire setup of IApp. Enterprise Organization along with its set of Catalogs drives the different functional pieces in IApp.



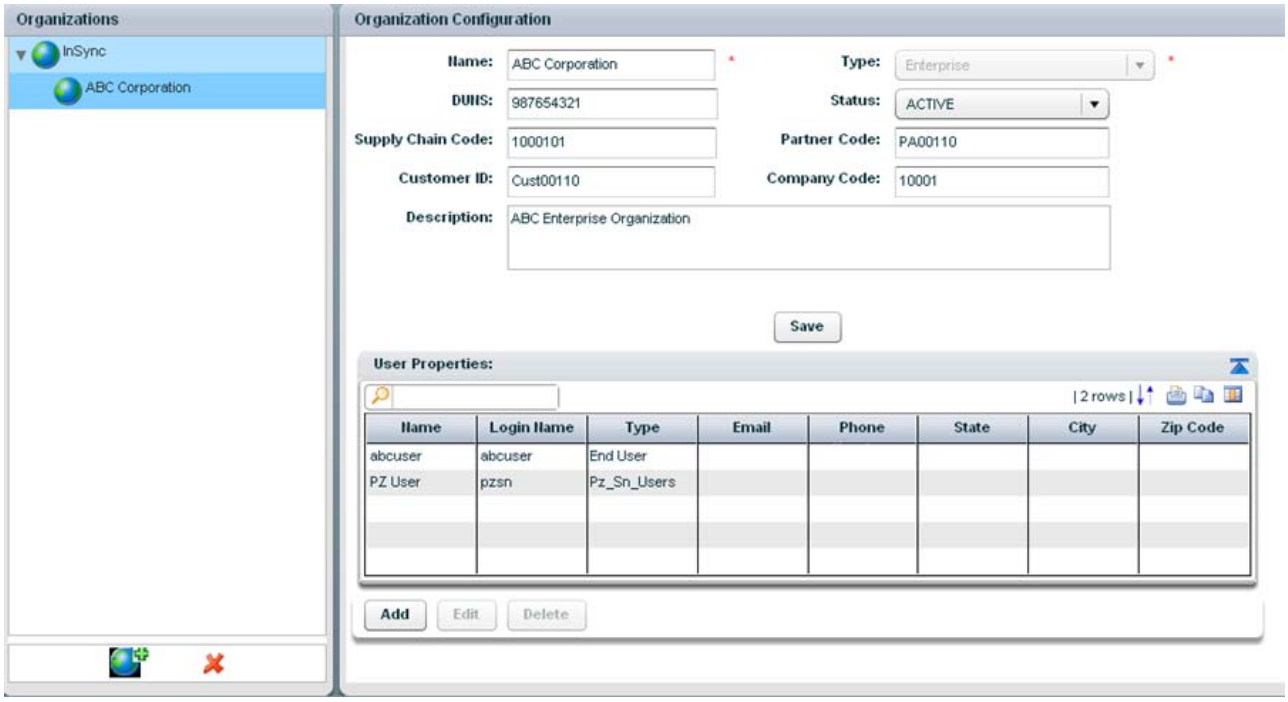
To create an Enterprise Organization select the Home Enterprise and click on



Attributes for the Enterprise are

Attribute	Description
Name	Organization name
Type	Type of the enterprise organization. By default IApp provides Enteprrise / Customer / Supplier at the Home enterprise and Customer / Supplier at the Enterprises created at Home. Note that the Organization type structure is controlled by the <EnterpriseOrg_OrgType> catalog. This catalog gets created once the Enterprise is created. To be able to create more organization types, the types have to be added to the catalog to be able to create the organization.
DUNS	The Data Universal Numbering System (DUNS) number is a unique nine character identification number provided by Dun & Bradstreet (D&B). Every Business entity will have a unique DUNS number
Supply Chain Code	Code used for supply chain management
Partner Code	Code for the partner
Customer Id	Code for the customer
Company Code	Organization code
Description	Short Description

Click ***'Save"*** to create the Enterprise Organization.



- Creation of Enterprise involves creating a set of catalogs required at that Enterprise to control its access. The catalogs created in the background are
    - <EnterpriseName>_EMS
    - <EnterpriseName>_OrgType
    - <EnterpriseName>_PersonType
    - <EnterpriseName>_Menu
- For more information on what each of the catalog means refer to Catalogs section
- The Users properties panel is disabled till the Enterprise organization is created.
- The type of the organization cannot be edited once the organization is created.

Note: Organizations of type 'Enterprise' can be created only at the Home Enterprise. Enterprise organization cannot be created at the other enterprises.

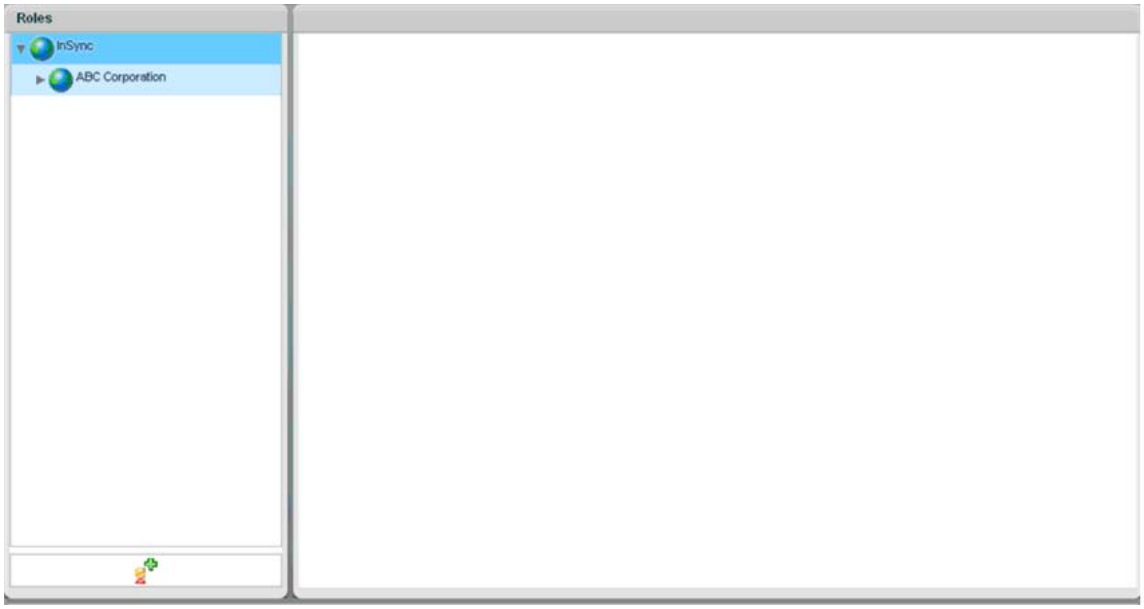
To create Organization of other Organization Types for the Enterprise Organization

- Select the Enterprise Organization and click on
    - The Types can be created from the catalog section and added to the <EnterpriseName>_OrgType catalog.
    - Follow the steps mentioned above.

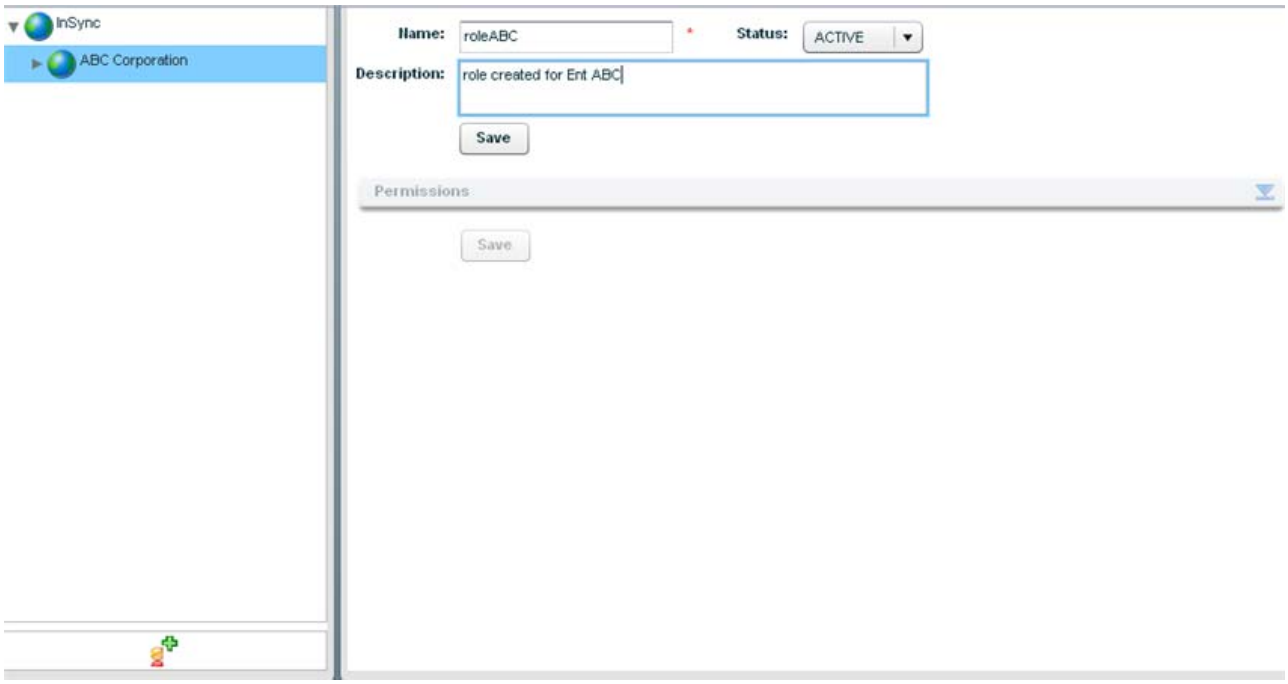
13.9 Roles and Permissions

The set of activities and the access to the Enterprise organizations for the user is based on the Roles and permissions assigned to the user at the time of User creation.

The roles are created by Enterprise Organization and can have a set of roles defined. The permissions are set to the role at the time of role creation.



To create roles select the Enterprise Organization and click



Attributes for creating a role are:

Attribute	Description
Name	Unique name for the Role within the Enterprise
Status	Active / InActive
Description	Short description

7. Click on ‘Save’ to save the role to the left side tree.
8. Existing roles are listed by Enterprise Organization in the left side panel.

Note: Permissions should be set for the created Role before assigning the role to the user.

Permissions define the access rights to the user through the Role. Permission is set for the roles which are assigned to the users.

Permissions are set on the role

- For the Enterprise organizations / Business locations that the user can have access to.
- For the menu access.
- For Assignable roles - roles that the user can assign to other users if they have the permission to create the users.

Note:

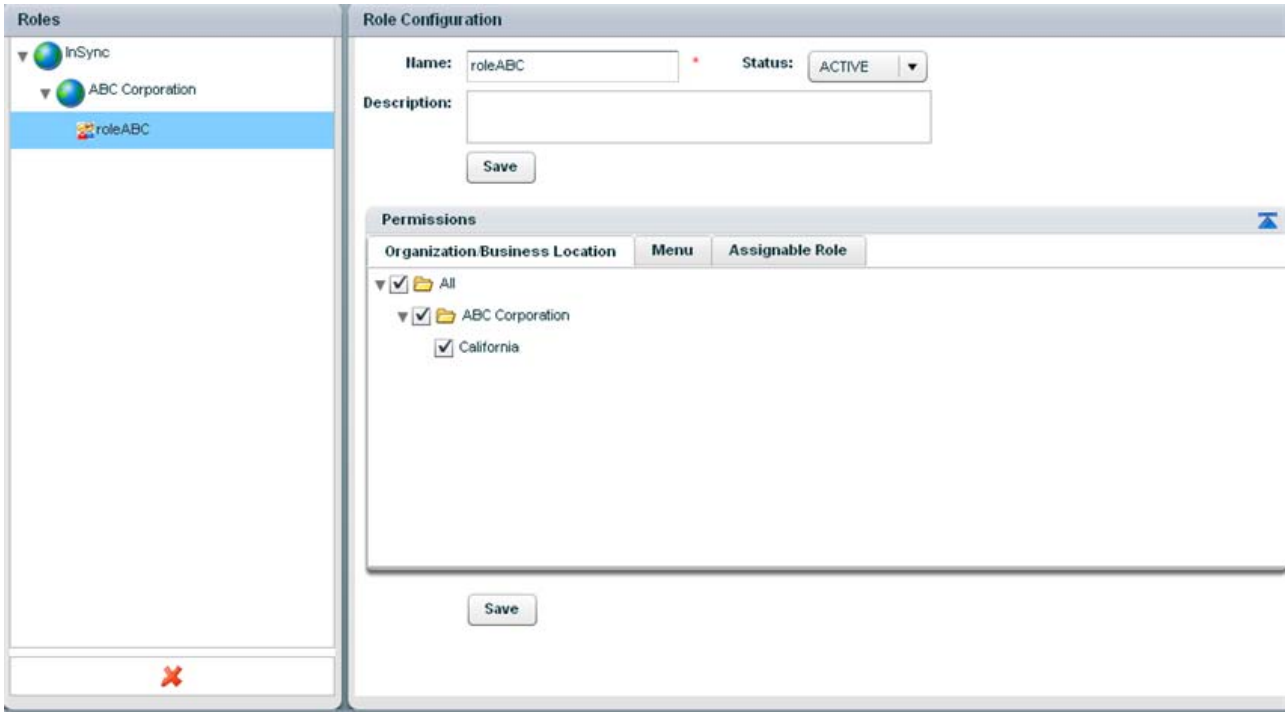
1. Users at the Home Enterprise can be given access to other Enterprises Organizations created within the Home Enterprise

2. Users at the Enterprises created within the Home Enterprise cannot have access to the Home Enterprise.

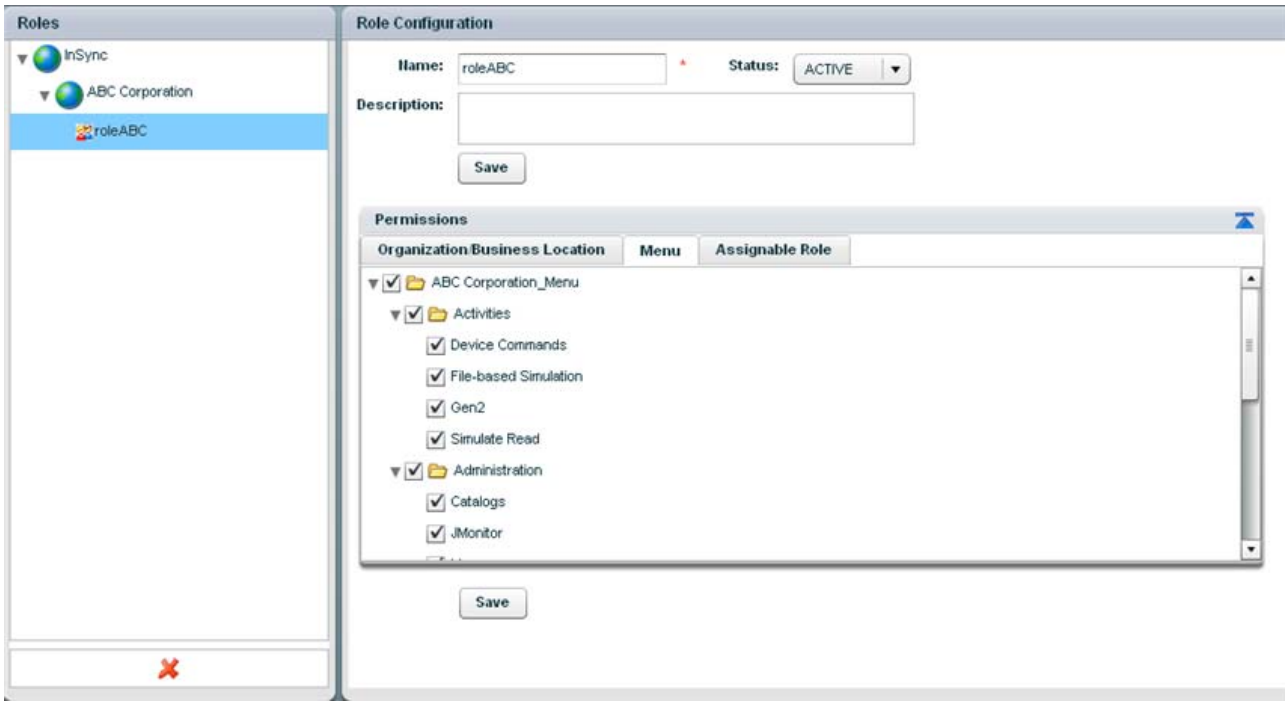
3. The Menu access to be given is picked up from the <Enteprise_Org>_Menu catalog.

To set the permissions

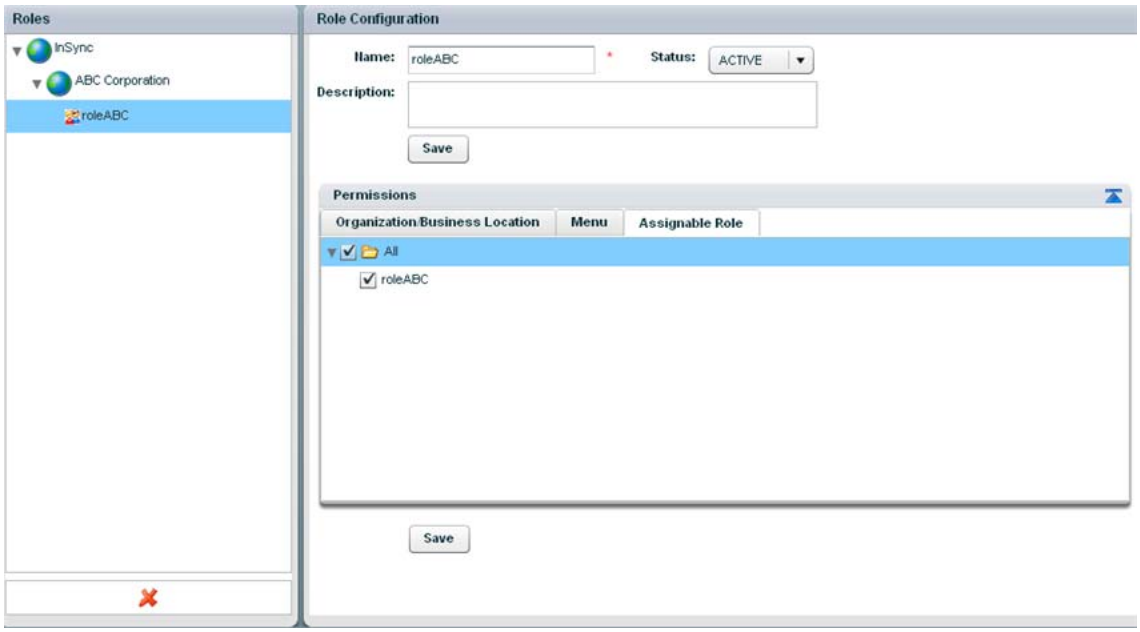
- Select the Role within an Enterprise
- Select the Organization / Business Location tab and expand the tree. Select / deselect the Organizations and business locations to be given access to



- Select the Menu tab and expand the tree. Select / deselect the menu items to which the access is to be given.



- Select the Assignable Role tab and expand the tree. Select / deselect the roles that the user can assign to other users

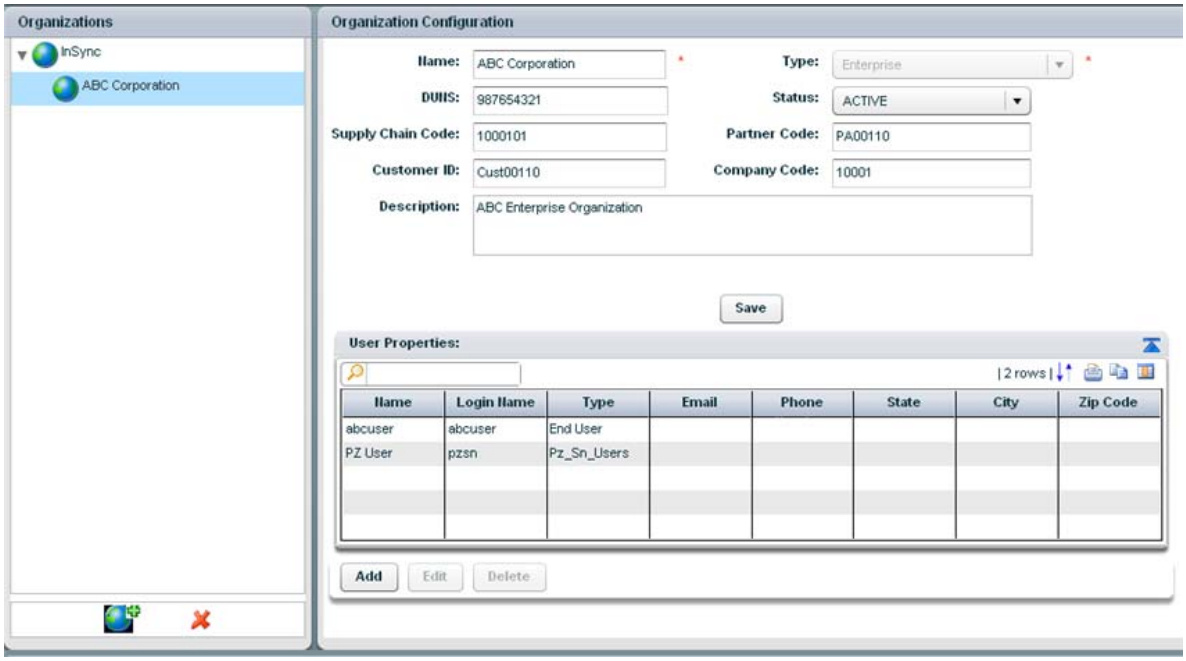


- Click '***Save'***.

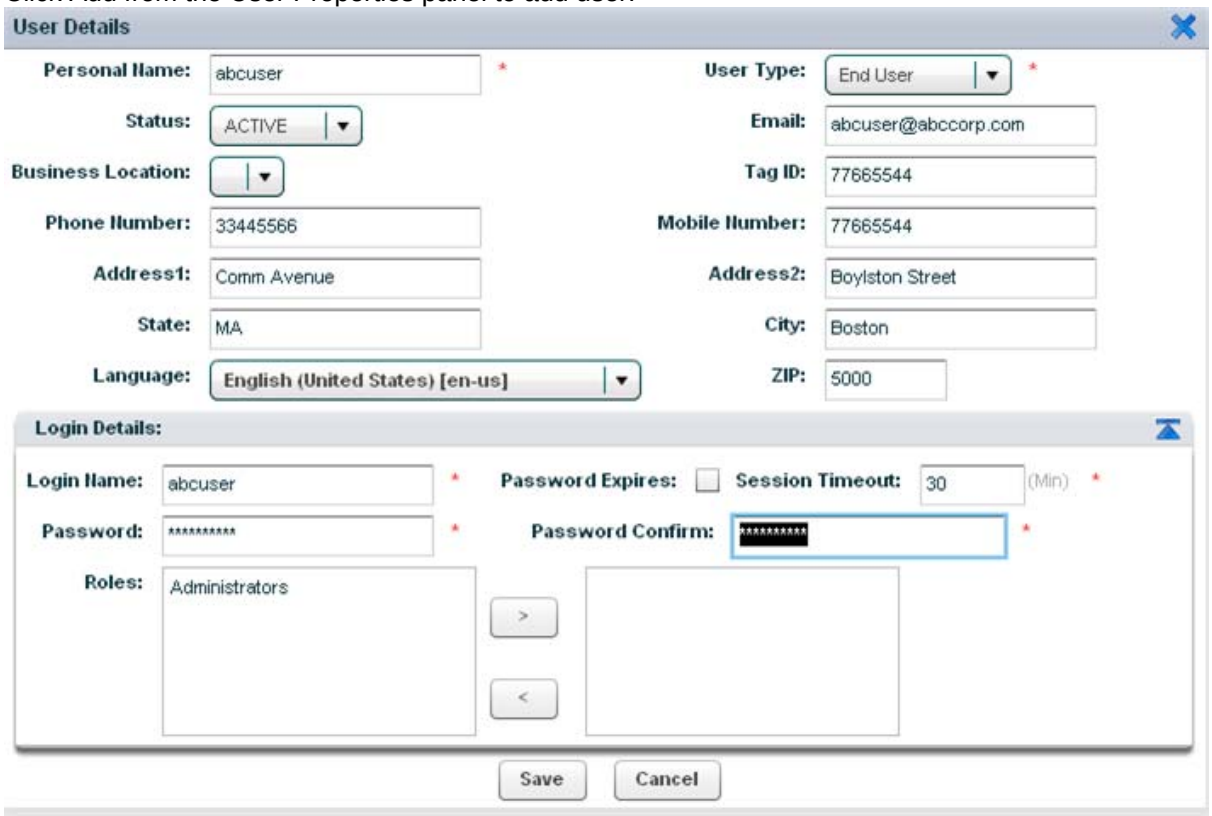
Note: It is recommended that access to the Administration and Software Menu Items be restricted to the Administrator only.

13.10 Users

Users in IApp are created from the Organization page and created for each Enterprise.



Click Add from the User Properties panel to add user.



Attributes for a user are:

Attribute	Description
Name	Name of the user
User Type	Type of the user getting created. By default Each Enterprise has End User, System and pz_sn_users created. Users of type pz_sn_users are required for process zone to communicate with sensornet. This user has to be configured in the config.xml of the process zone. Users of type System are required for all external calls to sensornet so that the requests can be authenticated. More types can be created from the catalogs page in <Enterprise_Org>_PersonType
Status	Active / InActive
Email	Email address of the user
Business Location	Location of the user
Tag ID	Tag Id assigned to the user.
Phone Number	Phone number of the user
Mobile Number	Mobile number of the user
Address1	Address details
Address2	Address details
State	State where the user belongs to
City	City where the user resides
Language	Language setting for the user. Note that this is for future use where the user can select the language he wants to view the Sensornet pages. Currently this setting is used in all customer applications built
ZIP	Zip code
User Details	
Login Name	Unique login name for the user in the Enterprise
Password	Password for the user
Password Confirm	Password confirmation
Password Expires	Check for the password to expire in 90 days. Uncheck to set the password to never expire.
Roles	The role to be assigned to the user. All the unassigned roles are listed in the left side and can be assigned / reassigned using the > and the < buttons.

- The ‘System’ type user is for external communication to Sensornet. All http / https calls to IApp uses the System user credentials.
- The pz_sn_user type user is for process zone to communicate to sensornet. This user has to be configured in the config.xml of the process zone
- The System and pz_sn_user users do not have access to Sensornet pages. This is driven by the attribute uiLogin set to false in the <Enterprise_Org>_PersonType catalog for the respective types.
- To edit the user select the user and click ***Edit***.

- To delete the user select the user and click ***Delete***

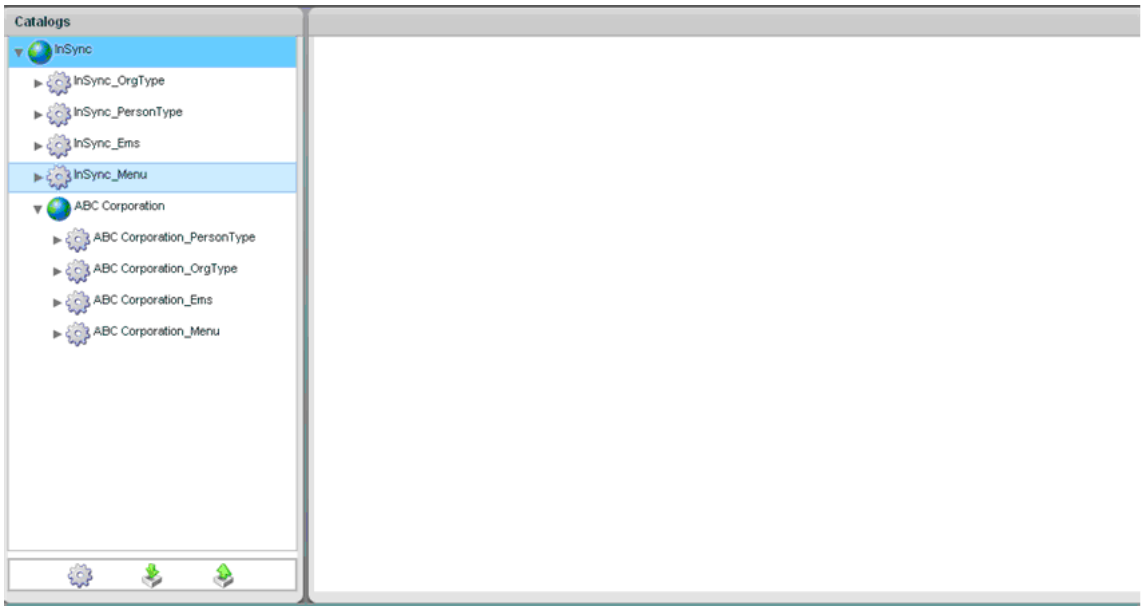
Note:

1. Any change in the password details should be communicated to the respective users and if it is a System user or a pz_sn_type user, changes should be made in the application where ever used so that the communication does not break.

2. The Administrator Role is available only at the Home Enterprise. There can be more than one user with the Administrator role.

13.11 Catalogs

Catalogs publish useful information for each enterprise and hold Meta data information at the Enterprise Organization level.



Each Enterprise organization created in IApp maintains the following catalogs.

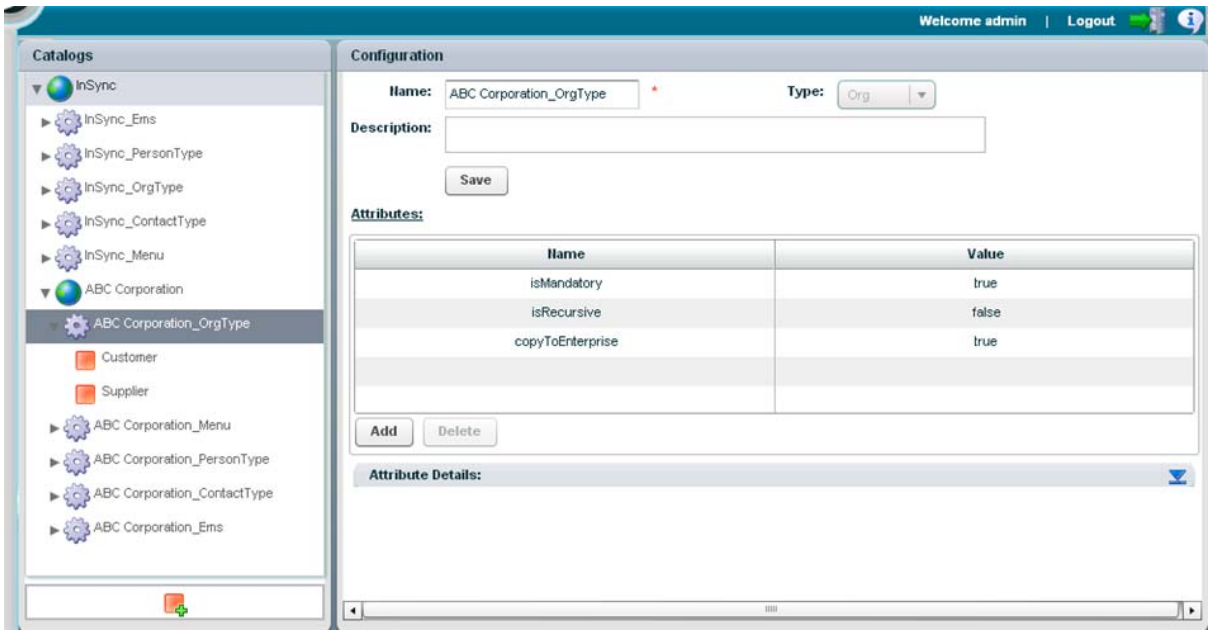
- **_OrgType**
- **_EMS**
- **_Menu**
- **_PersonType**

Note: Each catalog is prefixed with the Enterprise Organization name. It is required that the above mentioned catalogs are not tampered with by the administrator. Any changes to the catalogs mentioned will result in an unstable system.

Additional catalogs of different types can be created at each Enterprise organization for custom application purpose.

13.12 _OrgType

The different organization types are added in this catalog. The types added here get reflected in the Type drop down while creating the Enterprise

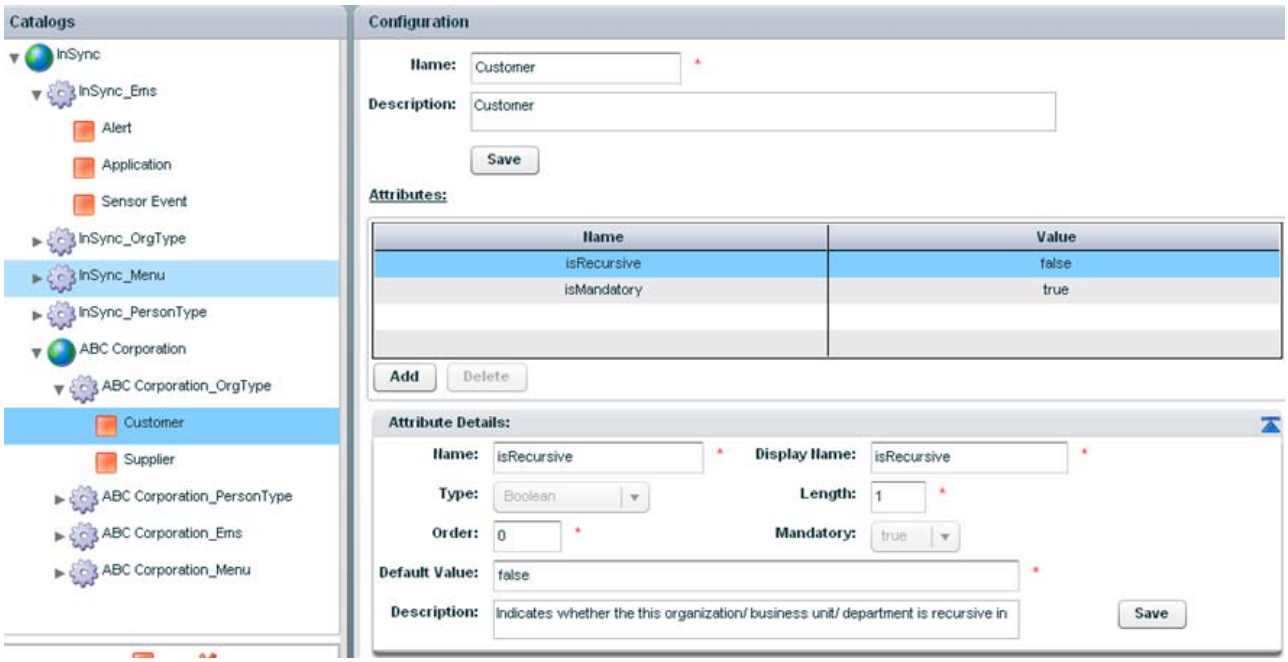


By default IApp provides Customer / Supplier organization types by default.

There are two attributes created by default for Org Type catalog. These are useful while creating the Organization from the Organization page.

- isRecursive: defines whether the organization of this type can repeat itself in the organization chart during the Organization creation.
- isMandatory: defines whether the organization of this type is mandatory to be created while creating the Organization tree during the Organization creation.
- copyToEnterprise: defines whether the catalog/category can be copied to the enterprise organization at the time of Enterprise organization creation. Note that this is applicable only at the Managed enterprise.

To edit the attribute select the attribute and change the values from the Attribute details panel. Click ***'Save'***



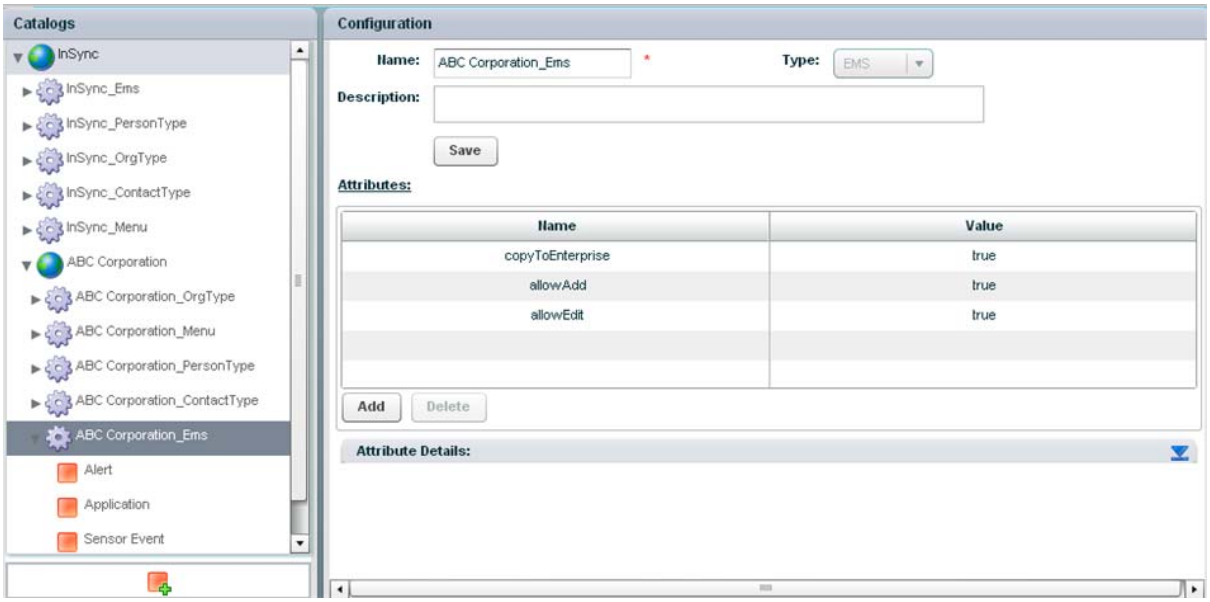
To add an attribute click ***'Add',*** *enter the details and click* ***'Save'***. The different attribute details will be explained in the **Add Catalog / Category section** in the later sections

Additional Attributes added will not have any impact unless used within the application

13.13 _EMSType

The _EMS type catalog refers to the different types of Events that can be configured in Event Management Section.

IApp provides Alert / Application and / Sensor Event by default.



EMS type catalog has the following attributes

- allowAdd: defines whether the users of the catalog within this enterprise can add more events / categories to this type

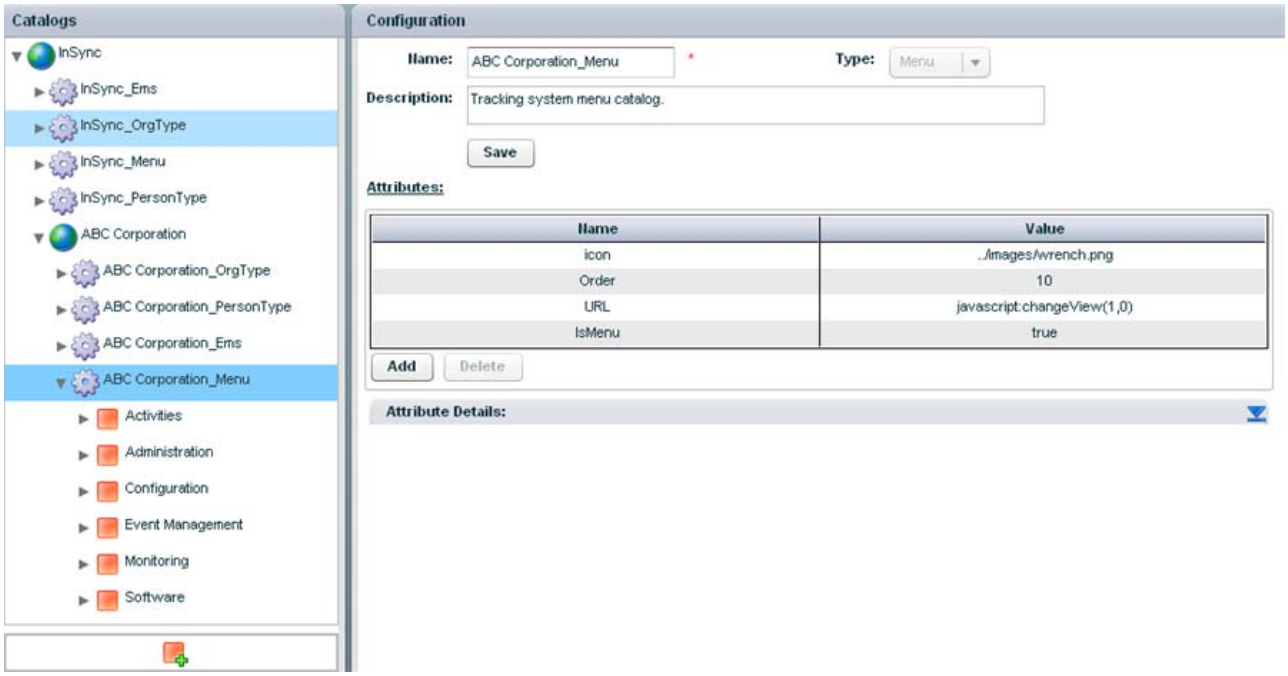
- allowEdit: defines whether the users of the catalog within this enterprise can edit the existing events of this type.
- copyToEnterprise: defines whether the catalog/category can be copied to the enterprise organization at the time of Enterprise organization creation. Note that this is applicable only at the Managed enterprise.

By default both the attributes for Alerts and Sensor Events are set to false where as Application type are set to true.

Note: Applications created here are visible in the Event Management Service for the Events to be configured. The system allows the creation of categories upto 2 levels only

13.14 _Menu

All menu activities (First level / second level menu items) are created in this catalog. The Menu catalog is useful while setting the permission for the Role in that Enterprise Organization.



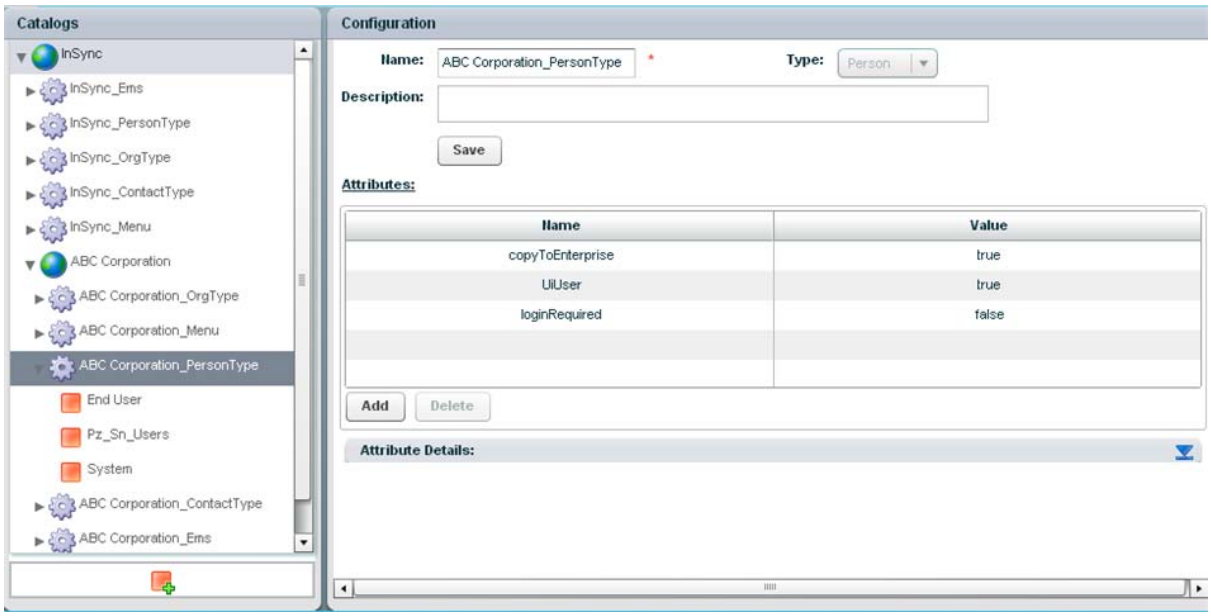
All Menu items created in the Menu catalog get rendered by IApp automatically provided the attributes are set properly.

Attributes provided by default are

- Icon – path to the image to be displayed on the menu for this menu item
- Order – order of appearance in the menu list
- URL – path to the source file
- IsMenu – decided whether the menu item is visible or not.
- copyToEnterprise: defines whether the catalog/category can be copied to the enterprise organization at the time of Enterprise organization creation. Note that this is applicable only at the Managed enterprise.

13.15 _PersonType

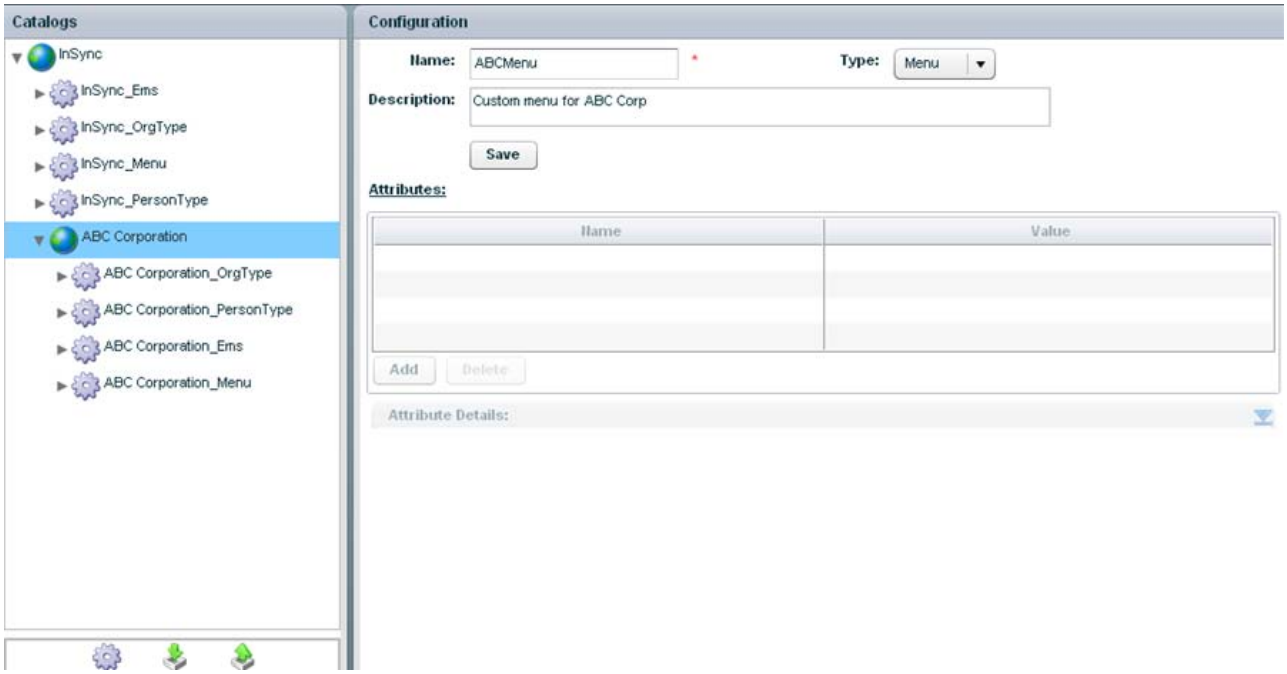
The different user types are created in this catalog. The types created here are shown in the User Type drop down during the creation of the User.



- IApp provides ‘End User’, ‘PZ_Sn_users’ and ‘System’ types by default.
- The Uiuser attribute decides if the user of the type can login to sensornet UI. If set to true then the user can login else cannot.
- The ‘loginRequired’ decides if the users of the type can make calls to sensornet from external applications. If the value is ‘true’ then the external requests are authenticated for the user.
- copyToEnterprise: defines whether the catalog/category can be copied to the enterprise organization at the time of Enterprise organization creation. Note that this is applicable only at the Managed enterprise.

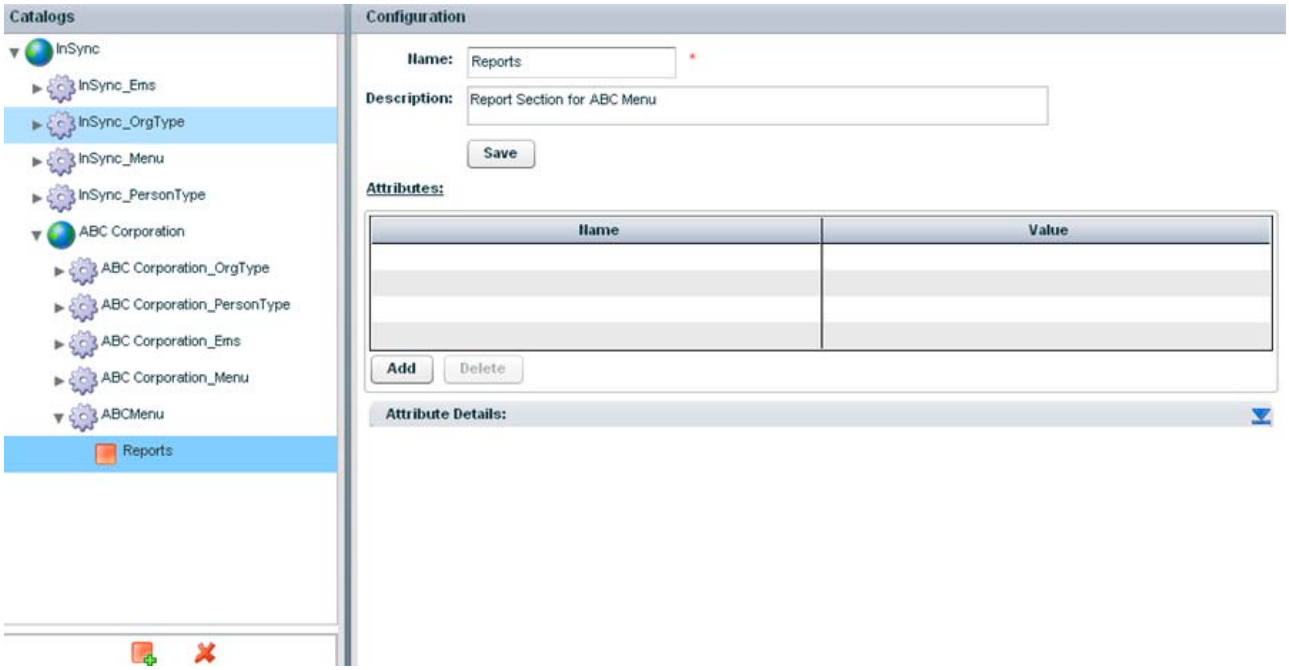
13.16 Add Catalog

To add a new catalog select the Enterprise Organization within which the catalog has to be created and click ‘ .



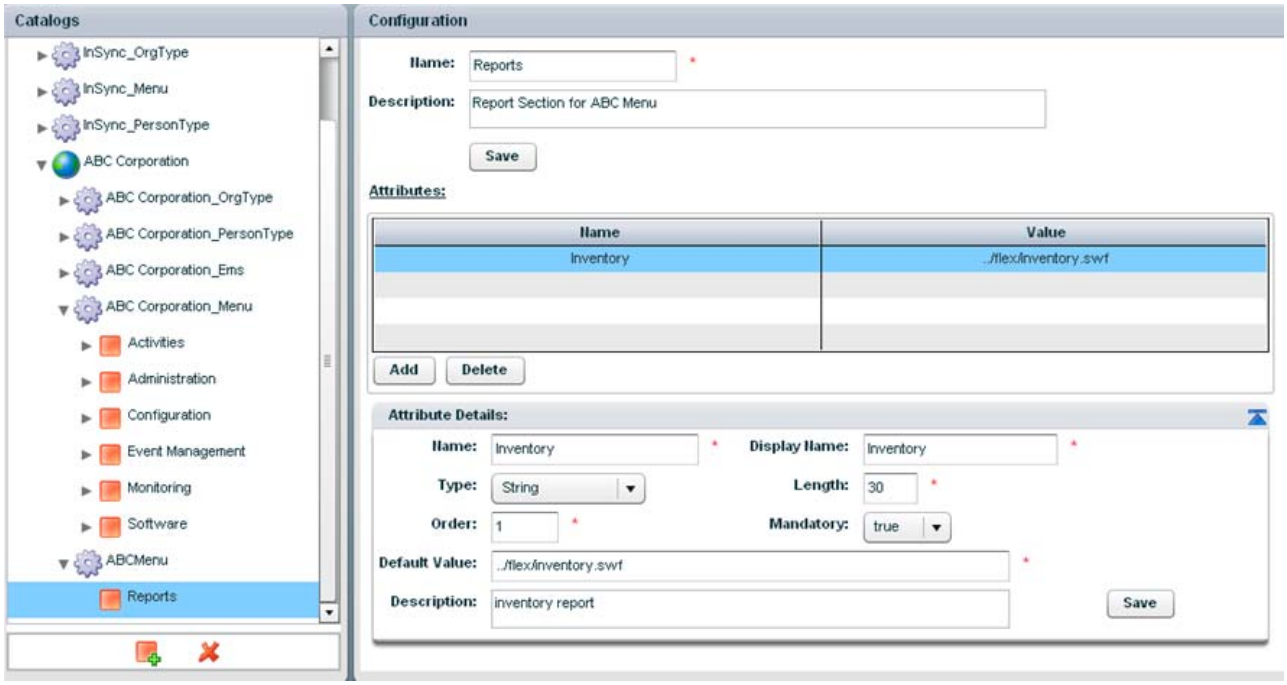
Attribute	Description
Name	Unique catalog name within the Enterprise Organization
Type	Select from Menu / EMS / Org / Person.
Description	Short description

- Click ‘Save’ to save the catalog. The catalog created gets added to the left panel tree.
- The Menu catalog gets created with the default attributes discussed in the previous sections.
- Multiple Items (categories) to the Menu catalog can be added in the same way by selecting the catalog and clicking button in the left lower panel



- The attributes panel gets enabled after the creation of the Catalog / the Category.

Click on ‘Add’ from the Attributes Panel to create an attribute.



Attribute	Description
Name	Attribute name
Display Name	Display name for the attribute
Type	This reflects the Default Value of this type

Length	Length of the Default value
Order	Display Order
Mandatory	True / False based on whether the Default Value is required or not.
Default Value	Value of the attribute. This should be of the Type selected and of the defined length
Description	Short description

13.17 Export / Import

Catalogs can be exported at each Enterprise Organization and imported to a different system

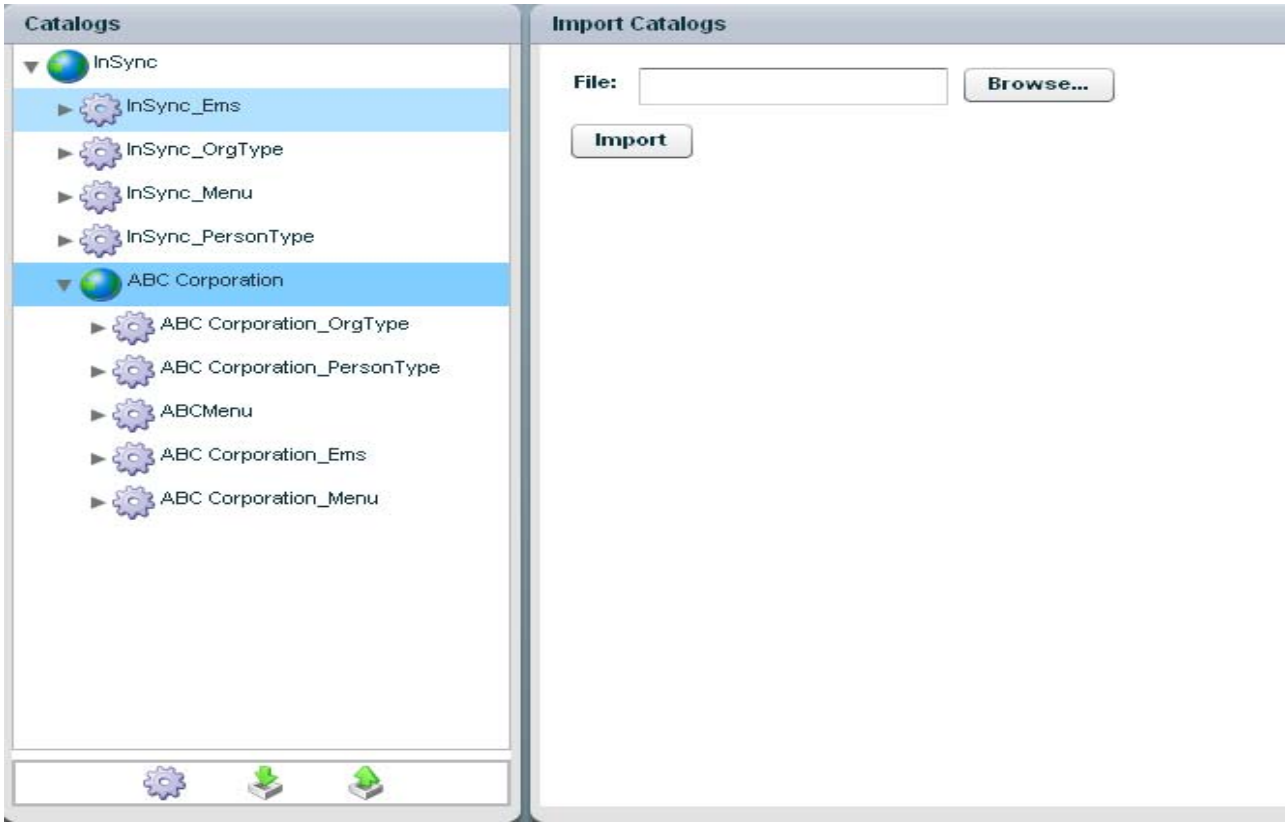
- Export at the Home Enterprise exports the Catalogs at all the Enterprise Organizations existing within the Home Enterprise
- Export at Enterprise Organization within the Home Enterprise exports Catalogs at that enterprise only.

Export

- Select the Enterprise Organization and click on
- Select the location to save the file and click Save
- File Catalogs.xml gets exported to the selected location.

Import

- Click in the left side lower panel

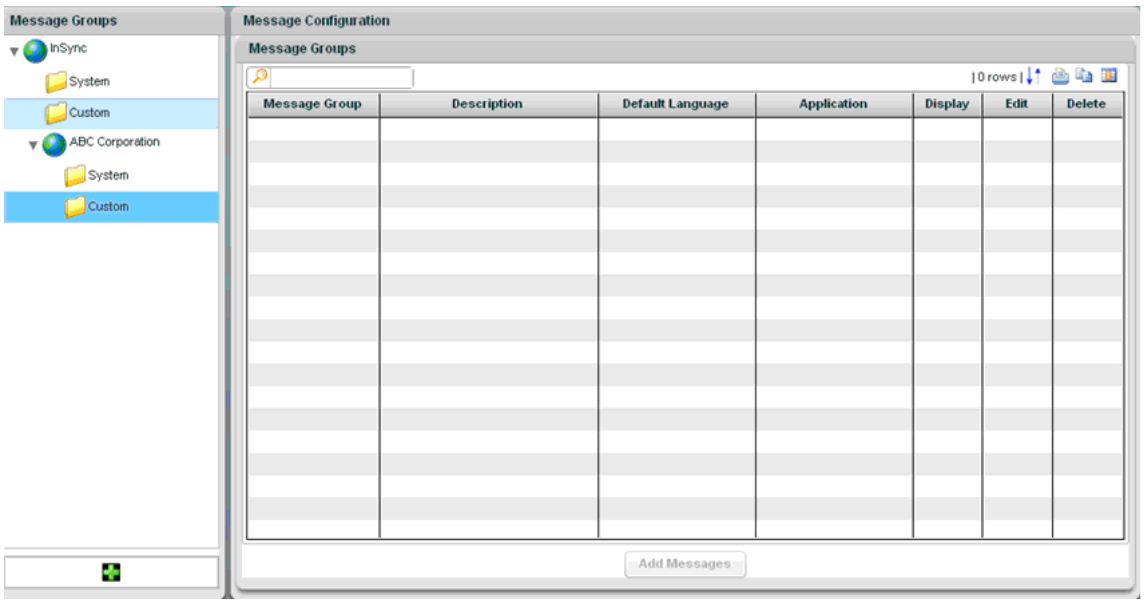


- Browse to the folder where the xml file to be imported is located
- Click Import button

Note: It is required that the Enterprise organization for which the Catalog import is happening and which exists in the .xml file exists in the system before the import happens.

13.18 Messages

Multilanguage support is provided in IApp using the Messages Feature. It works in conjuction with the underlying message framework that the applications built on top of IApp can take advantage of.



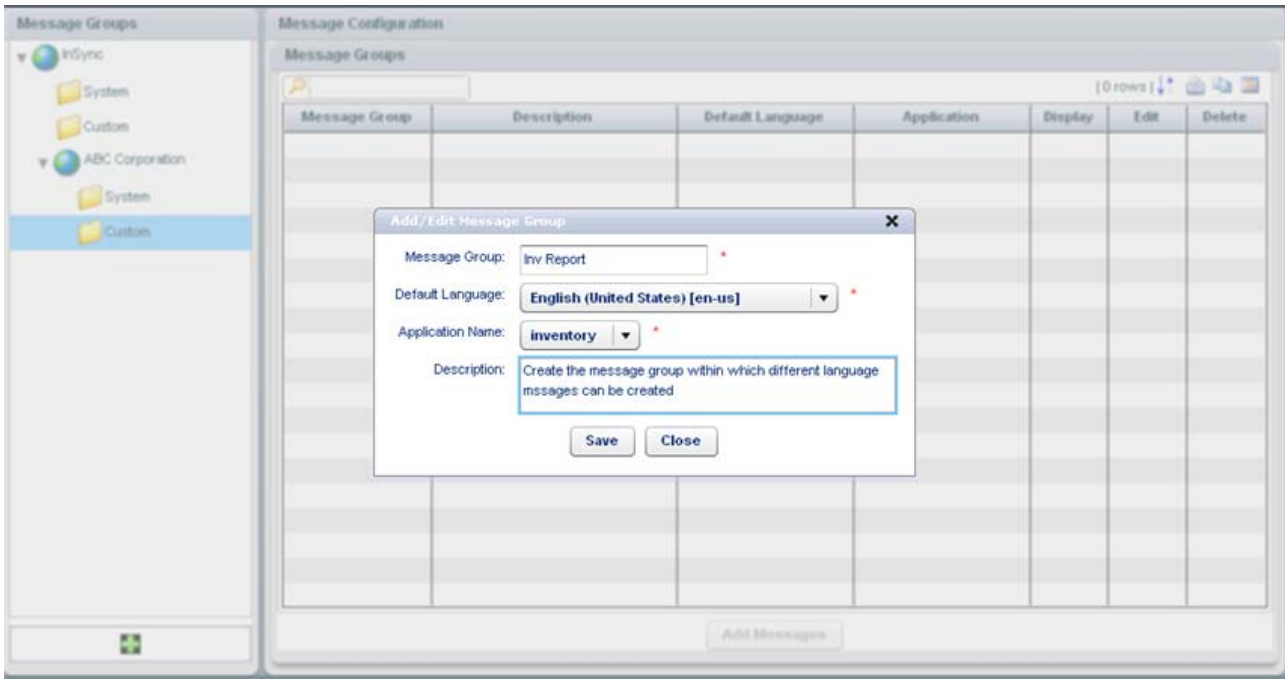
Messages are created by Enterprise Organization and can be segregated by System and Custom. Creation of messages involves

- creating message groups
- and messages for each of the group.

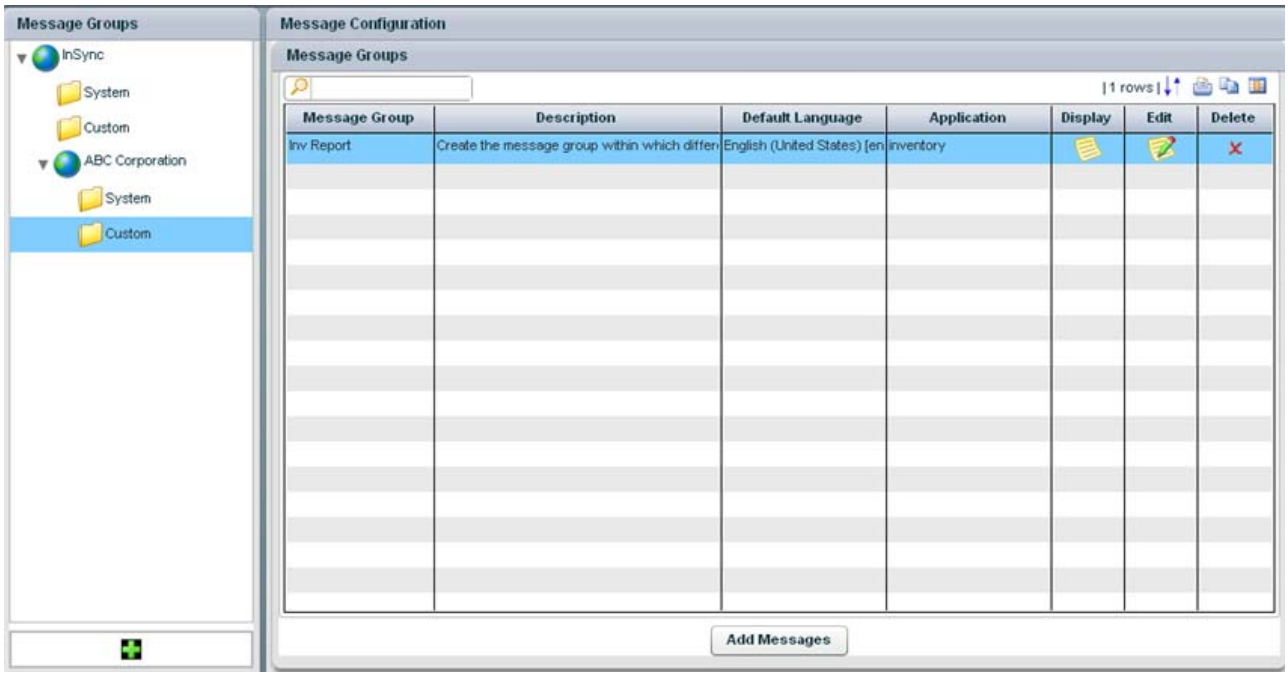
Note: The current version of IApp Sensornet UI does not use the multi language support provided by IApp. All applications built on IApp use the framework.

13.18.1 Add Message Groups

- Select the Enterprise Organization and select System / Custom
- Click ✚



Attribute	Description
Message Group	Brief name for the message group to be created
Default Language	Display language for the message group.
Application Name	Application name for which the group is getting created. The values in the drop down comes from the ct_application table.
Description	Short description for the message group



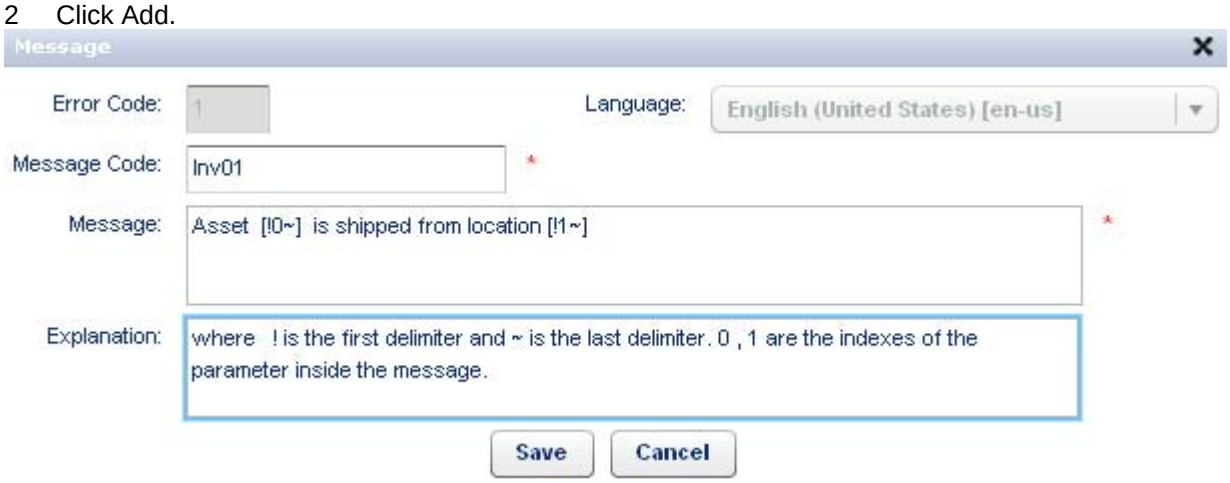
- Click to view the message group

- Click to edit the message group.
- Click to delete the message group.

13.18.2 Add Messages

1 Select the message group and click Add Messages.

2 Click Add.



3 By default the message will be in the default language of the message group.

Attribute	Description
Error Code	Sequence number generated by IApp
Language	language set for the message group
Message Code	Code by which the message can be retrieved from IApp

Message	Actual message to be used in the applications wherever necessary. Note that **!** is the first delimiter and **~** is the last delimiter. 0 , 1,2 …are used as indexes. The delimiter and the index are used together to create a variable which will be replaced at runtime by IApp.
Explanation	Brief description for the message

4 For multiple language support for the same message; select the message and click on ***Translation***

5 Click Add to add the message

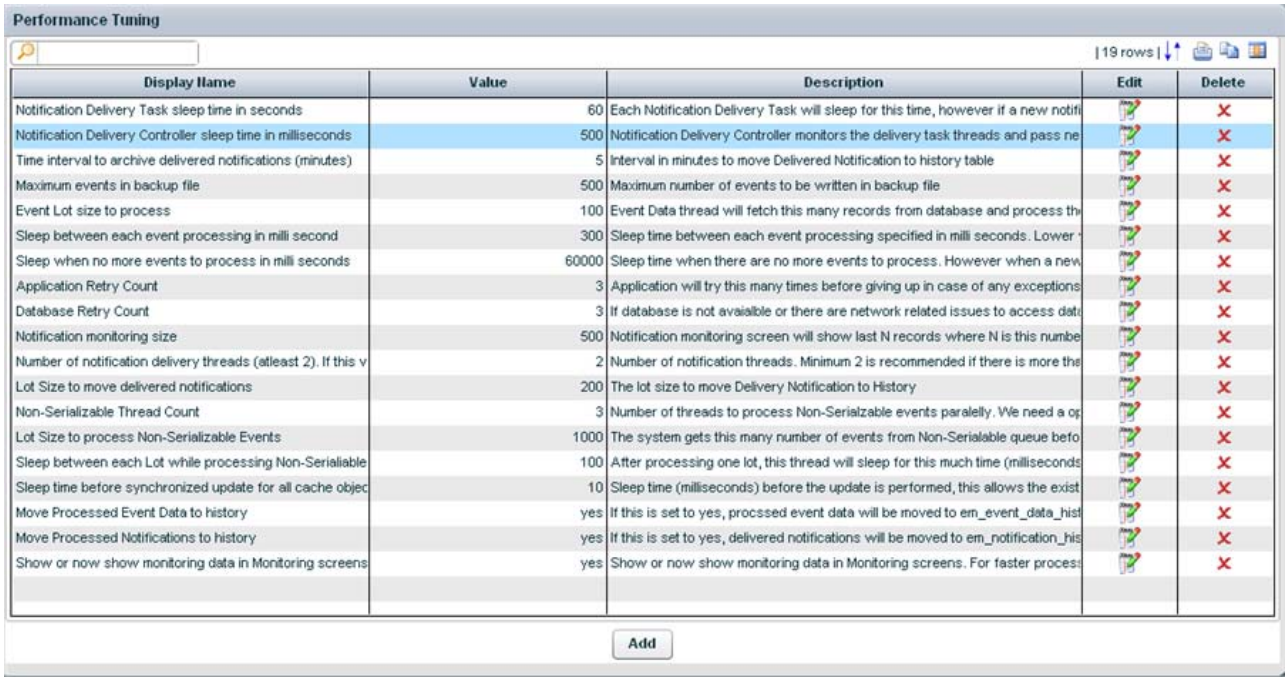
Attribute	Description
Error Code	Sequence number generated by IApp
Language	Select the language in which the message is getting created.
Message Code	Code by which the message can be retrieved from IApp. This is defaulted to the default language message code
Message	Actual message in the language chosen in the Language drop down to be used in the applications wherever necessary. Note that **!** is the first delimiter and **~** is the last delimiter. 0 , 1,2 …are used as indexes. The

	delimiter and the index are used together to create a variable which will be replaced at runtime by IApp.
Explanation	Brief description for the message

Note: For more information how to use the IApp Internationalization framework in the custom applications ask for Internationalization Support in IApp.doc.

13.19 Performance Tuning

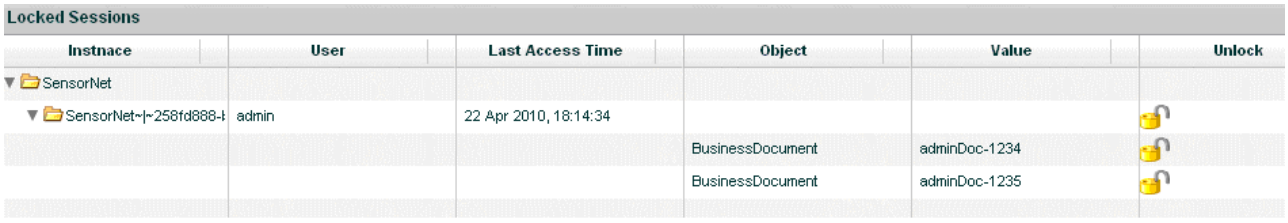
The performance tuning captures the different configuration parameters used by IApp during the processing. Changes to the Value of each attribute shown in the grid will affect the performance of the system.


Performance Tuning
19 rows
Display Name
Value
Description
Edit
Delete
Notification Delivery Task sleep time in seconds
60
Each Notification Delivery Task will sleep for this time, however if a new notifi
Notification Delivery Controller sleep time in milliseconds
500
Notification Delivery Controller monitors the delivery task threads and pass ne
Time interval to archive delivered notifications (minutes)
5
Interval in minutes to move Delivered Notification to history table
Maximum events in backup file
500
Maximum number of events to be written in backup file
Event Lot size to process
100
Event Data thread will fetch this many records from database and process th
Sleep between each event processing in milli second
300
Sleep time between each event processing specified in milli seconds. Lower
Sleep when no more events to process in milli seconds
60000
Sleep time when there are no more events to process. However when a new
Application Retry Count
3
Application will try this many times before giving up in case of any exceptions
Database Retry Count
3
If database is not avaialble or there are network related issues to access dat
Notification monitoring size
500
Notification monitoring screen will show last N records where N is this numbe
Number of notification delivery threads (atleast 2). If this v
2
Number of notification threads. Minimum 2 is recommended if there is more ths
Lot Size to move delivered notifications
200
The lot size to move Delivery Notification to History
Non-Serializable Thread Count
3
Number of threads to process Non-Serialzable events paralelly. We need a op
Lot Size to process Non-Serializable Events
1000
The system gets this many number of events from Non-Serialable queue befo
Sleep between each Lot while processing Non-Serialiable
100
After processing one lot, this thread will sleep for this much time (milliseconds
Sleep time before synchronized update for all cache objec
10
Sleep time (milliseconds) before the update is performed, this allows the exist
Move Processed Event Data to history
yes
If this is set to yes, procssed event data will be moved to em_event_data_hist
Move Processed Notifications to history
yes
If this is set to yes, delivered notifications will be moved to em_notification_his
Show or now show monitoring data in Monitoring screens
yes
Show or now show monitoring data in Monitoring screens. For faster proces
Add

Note: It is recommended that the Value is not changed without proper guidance and should strictly be an administrative user task who understands each of the parameters.

13.20 User Locks

IApp provides Locking service framework for applications to lock the objects in use during a transaction. The User Locks screen displays all locks in possession of the logged in user and allows the user to unlock the objects.



- Click to unlock the object.

13.21 Organization Locks

Organization Locks displays all locks acquired by Users in an Enterprise. All locks are displayed by Enterprise organization.

Select to unlock the objects.

Note: - Check IApp - LockingService.doc for more information on how to use the Locking framework in the applications.

iv. Section 4
iApp Advanced Capabilities

14 iApp Connection Objects

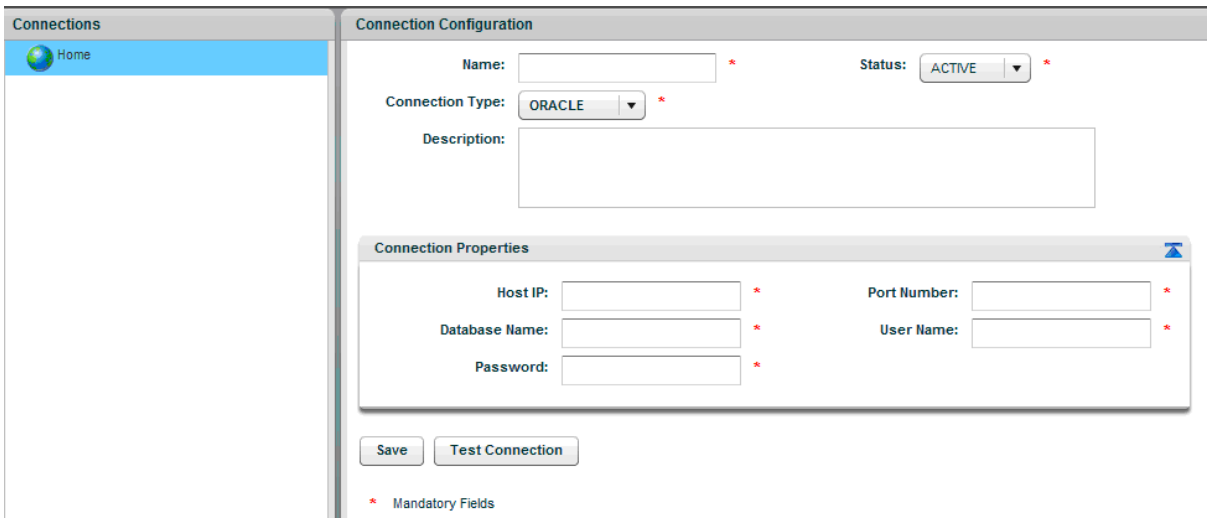
14.1 Introduction

We need to create a connection object to establish a physical connection to external entities like SAP, DataBase etc. These connection objects will capture the connection parameters and encapsulate a connection object that we can use to execute commands/queries. There is a need to invoke these objects from several places inside the application. We should NOT try to create same connection from several places n the application; this will result in improper connection object handling (like opening, closing, reusing connections).

Connection Object framework provides users/developers to create connection objects and use them across the application. They need not worry about the connection properties, opening, closing the connections. The framework will take care of these needs. Also since the connection properties can be modified from UI, there is no need to store them in cryptic xml, properties files and most importantly would not require server restart if we change any of the connection properties

14.2 Connection Types

Right now we support connection types for DataBase (Oracle, DB2, SQL Server, MySQL), SAP, SMTP. These types are implemented in the product and users can create connections for these types. Go to Software, click on the organization, click on Add Connection icon at bottom. Based on the connection type, appropriate connection properties are displayed. You can give a name to this connection and use this inside the application whenever you want to connect to this entity



14.3 Example:

In above section you have created connection, in this section we will try to use the connection to access the database.

14.3.1 To get connection object

```
com.insync.web.flex.service.ConnectionManager connectionManager  =
com.insync.web.flex.service.ConnectionManager.getInstance();

Connection conn =
(Connection)connectionManager.getConnection("TestConnection", _orgKey);
```

Where MyConnection is the name of the connection created in the above screen, _orgKey is the enterprise organization key where the connection is created.

14.3.2 To get OrgKey?

It depends on where you are writing the code. There are three areas where you might want to create connections and use them
1) UI Services -- if you are serving client requests from UI
2) Handlers -- if you are processing a event handler
3) Jobs -- if you are in a custom job implementation class

14.3.3 In the case of UI Services :

```
Personal user = (Personal) map.get(SessionSchema.USER);
BigDecimal orgKey = user.getOrgKey();
```

In the case of HANDLER

```
this._orgKey
```

Note: There is _orgKey existing in the super class that is BaseEventHandler

In the case of Jobs :

```
JobDetail job = context.getJobDetail();
List orgDetails = (List)job.getJobDataMap().get("Org");
String orgKey = (String) orgDetails.get(0);
```

Complete listing is below.

```
package com.kiran.handler;
import java.math.BigDecimal;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

import com.insync.dataobject.framework.connection.ConnectionEntity;
import com.insync.web.flex.service.ConnectionManager;

public class TestConnection {

	public TestConnection()
	{

	}
	public void processConnection()
	{
		/*
		 * Obtain the org[[]]
```

```
                */

                BigDecimal _orgKey = null;

                /**  Procedure to get OrgKey in "UI Service , Event Handler and JOB"
                 * ----------------------------------------------------------------
                 * _____________________________________________________________
                 * |In the case of UI Services :                               |
                 * |---------------------------_                               |
                 * |Personal user = (Personal) map.get(SessionSchema.USER);     |
                 * |BigDecimal orgKey = this.getOrgKey(map, user);              |
                 * |____________________________________________________________|
                 *
                 *
_____________________________________________________________________________________
                 * |In the case of HANDLER :                                     |
                 * |------------------------                                    |
                 * |this._orgKey                                                |
                 * |                                                            |
                 * | Note :There is _orgKey exists in the super class  that is BaseEventHandler |
                 *
|____________________________________________________________________________________|
                 *
                  _____________________________________________________________
                 * |In the case of UI Services :                               |
                 * |---------------------------_                               |
                 * | JobDetail job = context.getJobDetail();                    |
                 * | List orgDetails = (List)job.getJobDataMap().get("Org");     |
                 * | String orgKey = (String) orgDetails.get(0);                |
                 * |____________________________________________________________|
                 */

                //In this example we are assuming orgKey as 1
                _orgKey = new BigDecimal(1);

                try
                {

        /*************************************************************************************************************
****
                                |

                                                |
                                |ConnectionEntity                   connectEntity                   =
connectionManager.getConnectionForOrganization([Connection Name], _orgKey); |
                                |Connection                          conn                           =
(Connection)connectionManager.getConnection(connectEntity);                 |
                                |                                                                   |

*************************************************************************************************************
                                */

                                ConnectionManager           connectionManager                       =
ConnectionManager.getInstance();
                                ConnectionEntity                  connectEntity                     =
connectionManager.getConnectionForOrganization("TestConnection", _orgKey);
                                Connection                         conn                             =
(Connection)connectionManager.getConnection(connectEntity);
```

```
                        PreparedStatement statement = conn.prepareStatement("SELECT
MATERIAL_TYPE FROM CT_MATERIAL_TYPE");
                        ResultSet rs = statement.executeQuery();
                         while (rs.next()){
                       String materialType = rs.getString("MATERIAL_TYPE");
                       System.out.println(materialType);
                    }
                         statement.close();
                         conn.close();

                }
                catch(Exception exe)
                {
                        exe.printStackTrace();
                }


        }
}
```

---

15 iApp Custom Jobs

15.1 Introduction

Sometimes we need to execute a specific task at specific interval of time, like creating daily reports at 6 PM etc. Each task is implemented as job in iApp. The job will be executed at the specified interval. In iApp jobs are implemented using Quartz APIs. Quartz is a open source implementation, and is integrated with iApp.

15.2 Structure

15.2.1 Components of job

Job Implementation Class
Each job is implemented as a java class. It should extend com.insync.ems.scheduler.quartz.BaseJob. In this class we need to implement execute method

public void execute(JobExecutionContext context) throws JobExecutionException

Quartz engine will invoke execute method as per the schedule

Parameters
We can pass input parameters to the execute method so that they can be used at execution time. These parameters have to be declared using annotations

```
@JobParameterAnnotation(displayName=Constants.DISP_LOTSIZE, defaultValue =
"100", mandatory = Constants.PROP_MANDATORY_TRUE, valueMethod = "",
exampleValue = "", toolTipValue = "This is mandatory")

public static String PROP_LotSize=Constants.PROP_LOTSIZE;
```

where displayName is the name given for the parameter to be shown in the UI

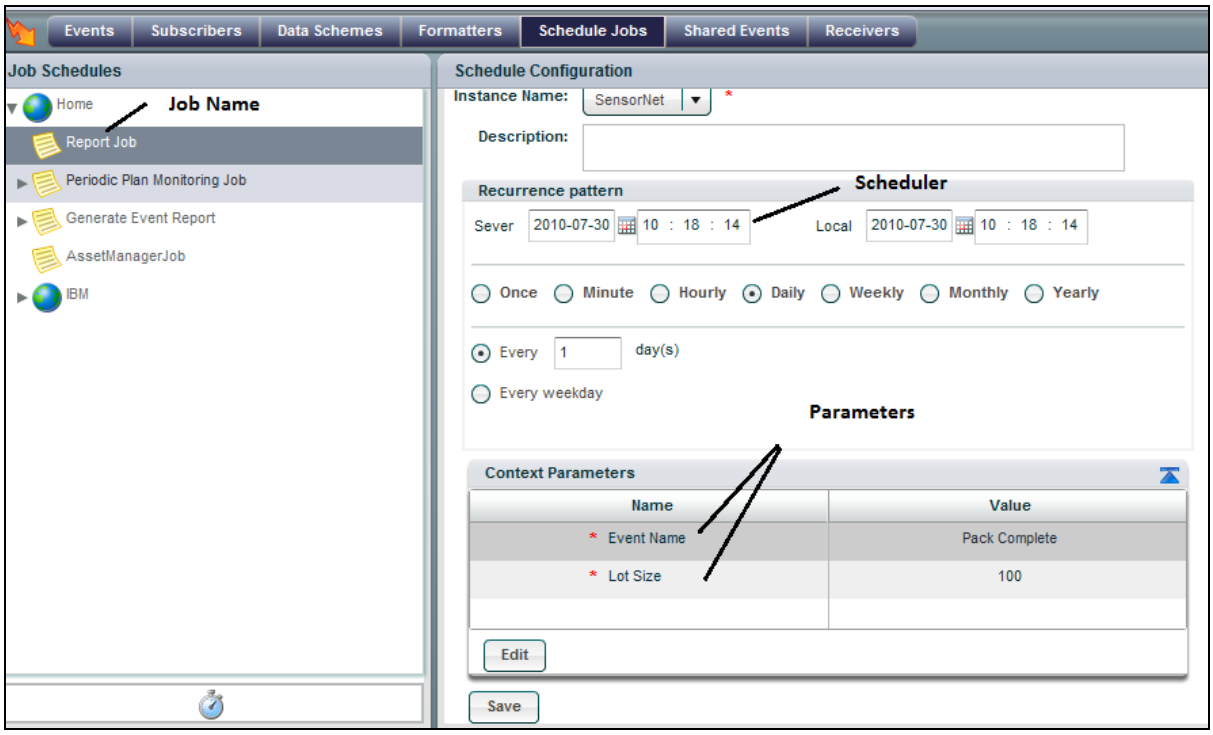
defaultValue is the value used by the code if user doesn't select any specific value from UI

mandatory tells whether it is required or optional

valueMethod is the implementation method to get the list of values to be shown in the drop down for user to select one

toolTipValue is the description of the parameter as seen when user hovers the mouse over the parameter

Scheduling
Jobs can be scheduled at specific interval using iApp scheduler in the Schedule tab



15.3 Example

Check the supported java version in the installer document, make sure you use this version only, otherwise it will result in unexpected behaviour

Copy ems.jar, quartz-1.5.1.jar into the build path so that appropriate classes are available for import, you can get these jar files from app/WEB-INF/lib

15.3.1 Create Class

Create a new java class using these guidelines

Class name must end with Job.

Good Examples : ReportJob.java, GenerateEmailJob.java

Bad Examples : CreateReport.java, GenerateEmail.java

Class must extend com.insync.ems.scheduler.quaurtz.BaseJob

Package name should not start with org.quartz

Add the following imports

- import java.util.List;

- import org.quartz.JobDetail;
- import org.quartz.JobExecutionContext;
- import org.quartz.JobExecutionException;
- import com.insync.ems.common.Constants;
- import com.insync.ems.scheduler.quartz.BaseJob;
- import com.insync.ems.util.JobParameterAnnotation;

15.3.2 Create parameters

@JobParameterAnnotation(displayName=Constants.DISP_EVENTNAME, defaultValue = "", mandatory = Constants.PROP_MANDATORY_TRUE, valueMethod = "com.insync.web.service.EventExplorerUIService.getEventIdNames", exampleValue = "", toolTipValue = "This is mandatory")

public static String PROP_EventName=Constants.PROP_EVENTNAME;

@JobParameterAnnotation(displayName=Constants.DISP_LOTSIZE, defaultValue = "100", mandatory = Constants.PROP_MANDATORY_TRUE, valueMethod = "", exampleValue = "", toolTipValue = "This is mandatory")

public static String PROP_LotSize=Constants.PROP_LOTSIZE;

make sure these variables are declared as public and their name start with “PROP_”, so that they are available to other packages to be displayed in UI

15.3.3 Override execute method

We can retrieve the job parameters using JobExecutionContext object as shown here

```
@Override
      public void execute(JobExecutionContext jec) throws
JobExecutionException {

try{
            JobDetail job = jec.getJobDetail();

            // get event name prameter
            String eventName = (String) ((List)
job.getJobDataMap().get(Constants.PROP_EVENTNAME)).get(0);
```

```
            // get lot size parameter
            String lotSize = (String) ((List)
job.getJobDataMap().get(Constants.PROP_LOTSIZE)).get(0);

// TO-DO

}catch(Throwable e){
                  throw new JobExecutionException(e.getMessage());
            }

      }
```

15.3.4 Build the class

Compile the above class and create a jar file and deploy to the server (WEB-INF/lib). Restart the server. Login to sensornet and go to Software tab, Jobs, under your organization, click on Custom and click on 'Add Job' icon at the bottom. From Job Implementation drop down select the java class you have created now, give a name to the job and save it.

To schedule this job, go to Event Management , Schedule Jobs tab, click on your organization and click on the job name you have created just now, click on Create Schedule icon at the bottom and select proper schedule

Complete java file is below.

```
package com.rfid.edgeware.web.service;


import java.util.List;

import org.quartz.JobDetail;
import org.quartz.JobExecutionContext;
import org.quartz.JobExecutionException;

import com.insync.ems.common.Constants;
import com.insync.ems.util.JobParameterAnnotation;

/*
 * Copyright (c) 1999-2010 Insync Software, Inc.
 * 224 Airport Parkway Suite 550 San Jose, CA 95110
 * All rights reserved.
 *
 * This software is the confidential and proprietary information
 * of Insync Software, Inc. You shall not disclose such
 * Confidential Information.
 */

/*
 * This is class is used to process the orders as a job.
 * It should be scheduled so that, it can run in specified intervals as job.
 *
```

```
 * $Log: ProcessOrderJob.java,v $
 * Revision 1.1.2.1  2011/03/21 02:11:20  balabandla
 * java templates
 *
 *
 */
public class ProcessOrderJob extends com.insync.ems.scheduler.quartz.BaseJob{

	public static final String FileVersion = "$Revision: 1.1.2.1 $";

	/**
	 * Field to hold event name.
	 */
	@JobParameterAnnotation(displayName=Constants.DISP_EVENTNAME, defaultValue = "",
mandatory = Constants.PROP_MANDATORY_TRUE, valueMethod =
"com.insync.web.service.EventExplorerUIService.getEventIdNames", exampleValue = "",
toolTipValue = "This is event name")
	public static String PROP_EventName=Constants.PROP_EVENTNAME;

	/**
	 * Field to hold lot size.
	 */
	@JobParameterAnnotation(displayName=Constants.DISP_LOTSIZE, defaultValue = "100",
mandatory = Constants.PROP_MANDATORY_TRUE, valueMethod = "", exampleValue = "",
toolTipValue = "This is tooltip")
	public static String PROP_LotSize=Constants.PROP_LOTSIZE;

	/**
	 * This is the actual method contains business logic, which will be in specified intervals,
	 * when it is scheduled. The JobExecutionContext will hold the property values and
	 * will be available to this method when it is scheduled.
	 */
	@Override
	public void execute(JobExecutionContext jec) throws JobExecutionException {

		try{
				JobDetail job = jec.getJobDetail();

			// get event name parameter
				String eventName = (String) ((List)
job.getJobDataMap().get(Constants.PROP_EVENTNAME)).get(0);
				System.out.println("event name - > " + eventName);

				// get lot size parameter
				String lotSize = (String) ((List)
job.getJobDataMap().get(Constants.PROP_LOTSIZE)).get(0);
				System.out.println("lot size - > " + lotSize);

				//TODO - code here
```

```
                }catch(Throwable e){
                        throw new JobExecutionException(e.getMessage());
                }
        }
}
```

16 Locking Service

16.1 Locking Service Overview

iApp is an enterprise application, we will have several users taking action on same object simultaneously. We need to make sure the data is consistent even when concurrent actions are being taken on same object. This makes more sense the users are modifying the configuration data (like event, device, location etc) or processing a business transaction (like shipping against a delivery order etc). We need to make sure only one user can modify the configuration or process a transaction at a time. If another user is doing similar action on same object we need to make sure the first user completes his action before allowing the second user to complete his action

To achieve this we need to LOCK the object in the application. Whenever a request comes , we check if there is an existing LOCK on this object, if there is no LOCK , we will LOCK this object against this session and continue. If it's already locked by another session, then we will display a friendly message to the user to try after sometime. Please note we are locking against SESSION, not USER. In iApp same user can login from multiple browsers and have multiple sessions at same time. So the lowest unique entity for us is SESSION. The locks are persisted in database , this will make sure we don't need to redistribute the locks in cluster mode and also in case of handheld applications the locks are still available if server goes down and comes back. Please note we don't cache the locks in memory, we delete the lock from database when we unlock it.

Locking for configuration changes (like device, event, administration screens) is NOT yet implemented, but we have the infrastructure.

Locking the business processes is typically implemented at the application layer. Let's take an example of shipping against a delivery order. Each delivery order may have one or more line items and each line corresponds to certain quantity of a material. The object we are locking in this example is delivery order. That means at any time only one operator can ship the entire document. We could also lock at line level , then same line cant be processed by more than one operator at any time. For simplicity we will assume we are locking the entire document. In our data model this corresponds to ct_business_document. When user is ready to ship a document, we will check if there is any lock on this document, if there is no lock we will lock this document against this session and continue. If there is already a lock on this document (by another session), then we will give a message to the user that this is being processed by another user. After user completes the shipment and confirms it, then we release the lock so that it's available to others. In some applications we can automatically release the lock when user navigates to another page so that it's available to other user. On logout or session timeout we release the lock automatically. However for handheld applications we don't release lock on logout, session timeout, we will need an explicit action/button click to release the lock. This is designed like this since handheld can run in offline mode, the synchronization service doesn't bother about logouts, timeouts.

We display the existing locks in Administration → User Locks screen, this will display the locks owned by THIS user. We also have another screen Adminitration → Organization Locks which will display all the locks owned by all users in this organization. We have the capability to unlock the object(s) from this screen (if you have access to this page). This page displays the objects locked in each session, along with user name, locked date etc. You can release only one object from the session even if there are more than one. This screen should be used as the last resort to unlock the objects as this overrides the user action. Its always preferable to unlock explicitly after the transaction is completely processed

Session id is generated by us at the time of login and this is used to store the locks. This is a combination of instance name and a UUID. We don't use http session id as that's not guaranteed to be unique. In case of handheld applications we use the device id as session id, this will make sure only one user can operate one handheld at any time. We need instance name to identify where it's originated. In cluster mode when one node goes down, all the transactions/locks created in that node/instance will be deleted and user has to redo the whole transaction. In cluster mode, we don't redistribute the transactional data, we do only configuration data, that's why will need to redo the whole transaction if the current node goes down in the middle of transaction.

16.2 Data model

Table name : LOCKED_OBJECTS

Column_name	Description	Example	
LOCK_KEY	System generated number	40074	System generated at creation time
SAAS_ORG_KEY	Enterprise org key of this object	1	
SESSION_ID	Unique session id	SensorNet~|~14eff13d-ef62-4b7c-88ea-579303eb3867	Generated by system at login
LOCKED_OBJECT	The object class being locked	com.insync.iApp.context.entity.doc.BusinessDocument	This is the object class
LOCKED_OBJECT_NAME	The object name to be displayed in UI	Delivery Order	This will tell the user what is the object that is locked
LOCKED_OBJECT_ROWS	The actual object being locked	78455	Here we can lock all rows or a particular row. We are locking document number 78455
LOCKED_OBJECT_FIELD_NAME	The column used to determine the locked object	Document Number	We are using document number to lock
SOURCE_INSTANCE	Instance where lock is created	SensorNet-1	Used internally in cluster mode

16.3 LockingService

16.3.1 To lock

public boolean lockObject(final BigDecimal orgKey,final String sessionId,final Class refClass, final Object key,final String objectDisplayName, final String objectFiledDisplayName, final String releaseOnRestart)

orgKey -- enterprise org key of the object being locked

sessionId -- unique session id to own the lock. All UI Services will have this in user object, in other places create your own unique number and use

refClass -- this is the object class of the locked object. This refers to one of our business object like com.insync.iApp.context.entity.doc,BusinessDocument ,
com.insync.iApp.context.entity.trackable.TrackableItem. the reason we need Class instead of String is to make sure this is a valid BO in our datamodel

key -- the object key which will be locked. To lock all rows in that table, use * , to lock few rows pass comma separated values

objectDisplayName -- type of the object class. For example BusinessDocument can be Delivery Order or Planned Order or Pick List. So tell us whats the object type

objectFiledDisplayName -- this will tell what's the field used to determine the unique lock object. This could be document number, serial number, or combination of more than 1 field

releaseOnRestart -- this will tell whether lock should be deleted on restarting the server or not. In case the lock is owned by HandHeld user then this should be false, else true. Because Handheld can run in offline mode, so don't want to release the locks automatically.

16.3.2 To unlock

public boolean releaseLockFromObject(final BigDecimal orgKey,final String sessionId,final Class refClass, final Object key)

orgKey -- enterprise org key of the object being unlocked

sessionId -- unique session id that owns the lock. Use the same session id used to lock
refClass -- this is the object class of the locked object. This refers to one of our business object like com.insync.iApp.context.entity.doc,BusinessDocument ,
com.insync.iApp.context.entity.trackable.TrackableItem. the reason we need Class instead of String is to make sure this is a valid BO in our datamodel

key -- the object key which is locked. To unlock all rows in that table, use * , to unlock few rows pass comma separated values

16.4 Locking UI

16.4.1 User Locks

User Locks shows the listing of all locks acquired by the user on the different objects. iApp uses the locking framework to lock the session for all objects requested by the user and unlocks them accordingly.

If by some reason the the object do not get unlocked, the administrator can unlock the object using the Unlock button against the session

16.4.2 Organization Locks

Organization Locks shows the listing of all locks at the Enterprise level. iApp uses the locking framework to lock the session for all objects at the Enterprise level.

If by some reason the the object do not get unlocked, the administrator can unlock the object using the Unlock button against the session.

17 iApp Internationalization

17.1 iApp Internationalization Overview

While implementing I18N(Internationalization) we need to take care of the following

- Time Zone
- Locale/Language
- Date Time formats
- Number Formats
- Currency

So far, we have implemented Time Zone and Language

In SensorNet application we will derive the language property from the Logged in user object and the same will be used to display the text to the user both in UI and backend. While creating the user from Administration screen, we need to select the Language

Example :

- Personal user = (Personal) map.get(SessionSchema.USER);
- String languageCode = user.getLanguageCode();

Messages are categorized as follows

- ApplicationName MessageGroup Message
- Example : Retail Application Alerts Asset [!0~] is shipped from warehouse [!1~] but not yet received in store [!2~] .

In sensornet application we have two categories of Messages

- System --These messages can't be changed. These are meant for all applications and system defined
- Custom – These messages can be changed by respective organization.

We can not edit Message Group if it has at least one message defined. Messages can be parameterized to show runtime values by having placeholders in the message.

- Ex:- Asset [!0~] is shipped from location [!1~]
- At runtime, this will look like this
- Asset [301A787291BC891000000002] is shipped from location [Fab3]

Here ! is the first delimiter and ~ is the last delimiter, 0 , 1 are the indexes of the parameter inside the message.

While defining the Messages in the application, the "Explanation" field should be properly filled. We need to mention the detailed causes and resolutions for this message. User should be able to understand why this message is generate and what action he/she needs to take to resolve this issue

We have Language service to have the translated text across the languages and the following method will be used for the same.

etTranslatedText*(*String messageGroup, String messageCode, String language, List<String> messageValues)

- MessageGroup – The group of the message in the above hierarchy
- MessageCode – Message code used while creating the message.
- Language – derived from logged in user.
- messageValues – Parameters if any to pass the run time values

Example :

- List<String> params = new ArrayList<String>();
- params.add(item.getSerialNumber());
- params.add(item.getCurrentLocation());
- params.add(item.getNextLocation());
- String message = _languageService.getTranslatedText(“Retail Application, "NOT_RECIEVED",
- user.getLanguageCode(), params)

While working with transaction dates we need to persist the following:

- Transaction date time local -- time in the source timezone
- Transaction date time GMT -- time in GMT
- Time Zone from which the transaction happened -- source TimeZone

Example :

- ct_received_date_local Date
- ct_received_date_gmt Date
- ct_received_timezone varchar2(64)

If we are using date attribute columns like date_attribute1, we need to make sure we store all the 3 columns using date attribute,varchar attribute columns Date_attribute1 --- store local time

- Date_attribute2 -- store gmt date for date_attribute1
- Attribute1 -- store timezone for this date

However for audit dates, we can store only one date ie GMT.

We will use local time to display in reports. We will use GMT time to do calculations across the dates in different geographical regions, calculate elapsed times, to feed to jobs etc

UI needs to send , receive the Time zone along with the local date (Converted long millis) from UI to server and server to UI. We need to make sure the dates are displayed in the local timezone in which that event has occurred regardless of where the server is running and where the browser is opened. For example, a shipment is received in EST at Mar 15 2010 13:12:11, then we need to show the received time as Mar 15 2010 13:12:11 , even if the browser is opened in EST time zone or other time zones. Similarly whenever date values are sent from UI to server, we need to make sure the UI captures the time in that timezone where the transaction has occurred and that timezone. For example the same shipment is moved to staging area in EST Mar 15 2010 14:32:21, then we need to send this time and timezone EST to server regardless of where the browser is running. Operators can be working in any timezone while doing transactions for other timezones. This is very important. We

can not assume system timezone always. The date field will be sent as long millis instead of string but we need timezone also as a separate field. The same logic applies to handheld.

v. Section 5
Integrating iApp With External Systems

18 SOAP Framework

18.1 Introduction

iApp invoke remote methods using SOAP framework. iApp uses this extensively to communicate between ProcessZone and SensorNet. For example to send commands to device from SensorNet, we invoke SOAP calls to ProcessZone from SensorNet. Similarly if we want to get from context data from SensorNet to ProcessZone , we invoke SOAP calls. We are using Apache SOAP for this. The SOAP server is invoked through http servlet SOAPServerServlet (mapped as soaprouter in the url). Every request is authenticated (using Basic Authentication in http).

18.2 How to invoke the SOAP method remotely

In this example we will try to invoke a method in ProcessZone.

url http://localhost:8090/ShipZone/soaprouter

please change the servername (localhost), port number, zone name appropriately

URN SOAP:URI-ZONE-ShipZone

please change the zone name appropriately

method name sendCommand

input parameters String deviceName, String command
output parameter String (response will be success or the exception message)

Authentication

The SOAP Server expects the authentication string (in Base64 encoded form) in the http header. We use Basic authentication, the string format is Basic username||organizationname:password (pl note there is a space between Basic and the rest of the string), the name of the parameter to be put in header is Authorization, here is an example

In Base64 encoded form

Basic ZWRnZXdhcmV8fEhvbWU6MzVZb0ZPZ2JESDAxWnJsNXQ5NFhtemc5L0pnPQ==

When we decode this we get

edgeware||Home:35YoFOgbDH01Zrl5t94Xmzg9/Jg=

please change the username, organizationname, password appropriately

SOAP Message will look like this

```
<?xml version='1.0' encoding='UTF-8'?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<SOAP-ENV:Body>
<ns1:sendCommand xmlns:ns1="SOAP:URI-ZONE-ShipZone" SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<Parameter0 xsi:type="xsd:string">R1001</Parameter0>
<Parameter1 xsi:type="xsd:string">read</Parameter1>
</ns1:sendCommand>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Sample .Net Client Code

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Net;
using System.IO;

namespace ProcessZoneSimulation
{
   public partial class SimulateSensornet : Form
   {
      public SimulateSensornet()
      {
         InitializeComponent();
      }

      private void btnSend_Click(object sender, EventArgs e)
      {
         try
         {
            //Authorization
            txtResponse.Text = "";
            HttpWebRequest request = (HttpWebRequest)WebRequest.Create(txtURL.Text);
            HttpWebResponse response = null;

            string authorizationText = txtUserName.Text.Trim() + "||" + txtOrganization.Text.Trim() + ":" +
txtPassword.Text.Trim();
            authorizationText = EncodeTo64(authorizationText);
            request.Headers.Add("Authorization", "Basic " + authorizationText);
            ASCIIEncoding encoding = new ASCIIEncoding();
            byte[] data = encoding.GetBytes(txtRequest.Text);
            if (data.Length > 0)
            {
               request.Method = "POST";
               request.ContentType = "text/xml";
               request.ContentLength = data.Length;
               Stream newStream = request.GetRequestStream();
               // Send the data.
               newStream.Write(data, 0, data.Length);
               newStream.Close();
            }
            response = (HttpWebResponse)request.GetResponse();
            if (response != null && response.StatusCode.ToString() == "OK")
            {
               StreamReader reader = new StreamReader(response.GetResponseStream());
               if (reader != null)
                  txtResponse.Text = (string)reader.ReadToEnd();
            }
            response.Close();
         }
         catch (Exception ex)
         {
            txtResponse.Text = ex.Message;
         }
      }


      public string EncodeTo64(string toEncode)
      {
         byte[] toEncodeAsBytes = System.Text.ASCIIEncoding.ASCII.GetBytes(toEncode);
```

```
            string returnValue = System.Convert.ToBase64String(toEncodeAsBytes);
            return returnValue;
        }

    }
}
```

Sample Java Client Code

```
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.io.OutputStreamWriter;
import java.io.Writer;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLConnection;

import org.apache.log4j.Logger;

import sun.misc.BASE64Encoder;

/*
 * Copyright (c) 1999-2010 Insync Software, Inc.
 * 224 Airport Parkway Suite 550 San Jose, CA 95110
 * All rights reserved.
 *
 * This software is the confidential and proprietary information
 * of Insync Software, Inc. You shall not disclose such
 * Confidential Information.
 */

/*
 *
 * $Log: ZoneServiceProviderTest.java,v $
 * Revision 1.1.2.1  2010/07/21 03:24:17  balabandla
 * test client
 *
 *
 */

/*
 * This is a SOAP client to invike sendCommand method in ZoneServiceProvider
 *
 */

public class ZoneServiceProviderTest {

        public static final String FileVersion = "$Revision: 1.1.2.1 $";

        private static final Logger _logger = Logger.getLogger(ZoneServiceProviderTest.class);

        public final static String DEFAULT_SERVER
  = "http://10.0.1.17:8090/ShipZone/soaprouter";

        public  void sendDeviceCommand(){

                try {
            URL u = new URL(DEFAULT_SERVER);
            URLConnection uc = u.openConnection();
            HttpURLConnection connection = (HttpURLConnection) uc;

            connection.setRequestProperty("Authorization", "Basic " + encode("edgeware||Home" + ":"+
"35YoFOgbDH01Zrl5t94Xmzg9/Jg="));
```

```
                connection.setRequestProperty("CONTENT-TYPE", "text/xml");

                connection.setDoOutput(true);
                connection.setDoInput(true);
                connection.setRequestMethod("POST");

                OutputStream out = connection.getOutputStream();
                Writer wout = new OutputStreamWriter(out);

                StringBuffer msg = new StringBuffer();
                msg.append("<?xml version='1.0' encoding='UTF-8'?>\n");
                msg.append("<SOAP-ENV:Envelope xmlns:SOAP-
ENV=\"http://schemas.xmlsoap.org/soap/envelope/\" xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
\n");
                msg.append("xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\"> \n");
                msg.append("<SOAP-ENV:Body> \n");
                msg.append("<ns1:sendCommand xmlns:ns1=\"SOAP:URI-ZONE-ShipZone\" SOAP-
ENV:encodingStyle=\"http://schemas.xmlsoap.org/soap/encoding/\"> \n");
                msg.append("<Parameter0 xsi:type=\"xsd:string\">R1001</Parameter0> \n");
                msg.append("<Parameter1 xsi:type=\"xsd:string\">read</Parameter1> \n");
                msg.append("</ns1:sendCommand>\n");
                msg.append("</SOAP-ENV:Body>\n");
                msg.append("</SOAP-ENV:Envelope>");

                wout.write(msg.toString());

                wout.flush();
                wout.close();

                _logger.debug("----------- request -----------");
                _logger.debug(msg.toString());

                InputStream in = connection.getInputStream();
                int x = 0;
                                StringBuffer sb = new StringBuffer();
                                while((x = in.read()) != -1)
                                        sb.append((char) x);

                                _logger.debug("----------- response -----------");
                                _logger.debug(sb);
                in.close();

              }
              catch (IOException e) {
                      _logger.error("Exception "+e);
              }
            }

            public static String encode (String source) {
              BASE64Encoder enc = new sun.misc.BASE64Encoder();
              return(enc.encode(source.getBytes()));
             }

            public static void main(String[] args){
                      ZoneServiceProviderTest client = new ZoneServiceProviderTest();
                      client.sendDeviceCommand();
            }

}
```

19 iApp Web Services

19.1 About iApp Web Services

- Material Type/Material/Item Creation
- URL Creation for plotting tag on the image
- Post Tag Read and RTLS Tag Read Events

19.2 Introduction

This document describes the webservices provided in iApp for the following.

- Creation of Material Type, Material and Items
- Posting Tag read and RTLS Tag read event
- Generating the URL for plotting the tag on the image

Note: Succesful installation of iApp is required to execute the services. **It is useful to use a SOAP client or configure in eclipse to construct the service calls using the WSDL**.

The WSDL has all the service calls that can be made on the webservice as well as the response.

19.3 Creating Material Type/Material/Items

To view the wsdl file access the URL after running sensornet.

http://<ipaddress>:<portnumber>/sensornet/services/swItemService?wsdl

where

<ipaddress> is the hostname/ipaddress of the server running sensornet
<portnumber> is the port on which tomcat is running.

WSDL:

```
<?xml version='1.0' encoding='UTF-8'?><wsdl:definitions
name="ContextItemServiceWSService"
targetNamespace="http://service.web.context.edgeware.insync.com/"
xmlns:ns1="http://cxf.apache.org/bindings/xformat"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://service.web.context.edgeware.insync.com/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:types>
<xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://lang.java" xmlns:tns="http://lang.java"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:complexType name="Throwable">
<xsd:sequence />
</xsd:complexType>
</xsd:schema>
<xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
```

```
targetNamespace="http://util.context.edgeware.insync.com"
xmlns:tns="http://util.context.edgeware.insync.com"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:simpleType name="AttributesTypes">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="CREATE_ITEM" />
<xsd:enumeration value="TRACKABLE_NUMBER" />
<xsd:enumeration value="TAG_ID" />
<xsd:enumeration value="SERIAL_NUMBER" />
<xsd:enumeration value="ASSET_ID" />
<xsd:enumeration value="SYSTEM_ID" />
<xsd:enumeration value="ALTERNATE_TAG_ID" />
</xsd:restriction>
</xsd:simpleType>
</xsd:schema>
<xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://soap.xml.javax" xmlns:ns0="http://lang.java"
xmlns:tns="http://soap.xml.javax" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:import namespace="http://lang.java" />
<xsd:complexType name="SOAPException">
<xsd:sequence>
<xsd:element minOccurs="0" name="cause" nillable="true" type="ns0:Throwable" />
<xsd:element minOccurs="0" name="message" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
<xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://service.web.context.edgeware.insync.com/"
xmlns:ns0="http://composer.webservice.context.edgeware.insync.com"
xmlns:ns1="http://util.context.edgeware.insync.com" xmlns:ns2="http://soap.xml.javax"
xmlns:tns="http://service.web.context.edgeware.insync.com/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:import namespace="http://composer.webservice.context.edgeware.insync.com" />
<xsd:import namespace="http://util.context.edgeware.insync.com" />
<xsd:complexType name="ArrayOfString">
<xsd:sequence>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="string" nillable="true"
type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="saveAndUpdateMaterial" type="tns:saveAndUpdateMaterial" />
<xsd:complexType name="saveAndUpdateMaterial">
<xsd:sequence>
<xsd:element minOccurs="0" name="itemData" nillable="true" type="ns0:MaterialData" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="organizationName" type="xsd:string" />
<xsd:element name="userName" type="xsd:string" />
<xsd:element name="saveAndUpdateMaterialResponse"
type="tns:saveAndUpdateMaterialResponse" />
<xsd:complexType name="saveAndUpdateMaterialResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="ns0:MaterialComposer"
/>
</xsd:sequence>
</xsd:complexType>
<xsd:element name="deleteItemByAttribute" type="tns:deleteItemByAttribute" />
<xsd:complexType name="deleteItemByAttribute">
<xsd:sequence>
```

```
<xsd:element minOccurs="0" name="attributeType" nillable="true"
type="ns1:AttributesTypes" />
<xsd:element minOccurs="0" name="attributeValue" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="deleteItemByAttributeResponse"
type="tns:deleteItemByAttributeResponse" />
<xsd:complexType name="deleteItemByAttributeResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="ns0:ItemComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="itemQueryByAlternateTagIds"
type="tns:itemQueryByAlternateTagIds" />
<xsd:complexType name="itemQueryByAlternateTagIds">
<xsd:sequence>
<xsd:element minOccurs="0" name="alternateTagIds" nillable="true"
type="tns:ArrayOfString" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="itemQueryByAlternateTagIdsResponse"
type="tns:itemQueryByAlternateTagIdsResponse" />
<xsd:complexType name="itemQueryByAlternateTagIdsResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="ns0:ItemComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="getMaterialsByNumber" type="tns:getMaterialsByNumber" />
<xsd:complexType name="getMaterialsByNumber">
<xsd:sequence>
<xsd:element minOccurs="0" name="materialNumbers" nillable="true"
type="tns:ArrayOfString" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="getMaterialsByNumberResponse"
type="tns:getMaterialsByNumberResponse" />
<xsd:complexType name="getMaterialsByNumberResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="ns0:MaterialComposer"
/>
</xsd:sequence>
</xsd:complexType>
<xsd:element name="itemQueryBySystemIds" type="tns:itemQueryBySystemIds" />
<xsd:complexType name="itemQueryBySystemIds">
<xsd:sequence>
<xsd:element minOccurs="0" name="systemIds" nillable="true" type="tns:ArrayOfString" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="itemQueryBySystemIdsResponse"
type="tns:itemQueryBySystemIdsResponse" />
<xsd:complexType name="itemQueryBySystemIdsResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="ns0:ItemComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="itemQueryByHistory" type="tns:itemQueryByHistory" />
<xsd:complexType name="itemQueryByHistory">
<xsd:sequence>
<xsd:element minOccurs="0" name="tagId" nillable="true" type="xsd:string" />
```

```
<xsd:element minOccurs="0" name="startDate" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="endDate" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="arg4" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="itemQueryByHistoryResponse"
type="tns:itemQueryByHistoryResponse" />
<xsd:complexType name="itemQueryByHistoryResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="ns0:ItemComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="deleteMaterialType" type="tns:deleteMaterialType" />
<xsd:complexType name="deleteMaterialType">
<xsd:sequence>
<xsd:element minOccurs="0" name="materialType" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="deleteMaterialTypeResponse"
type="tns:deleteMaterialTypeResponse" />
<xsd:complexType name="deleteMaterialTypeResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true"
type="ns0:MaterialTypeComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="itemQueryByTrackableNumber"
type="tns:itemQueryByTrackableNumber" />
<xsd:complexType name="itemQueryByTrackableNumber">
<xsd:sequence>
<xsd:element minOccurs="0" name="trackableNumbers" nillable="true"
type="tns:ArrayOfString" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="itemQueryByTrackableNumberResponse"
type="tns:itemQueryByTrackableNumberResponse" />
<xsd:complexType name="itemQueryByTrackableNumberResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="ns0:ItemComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="itemQueryByAssetIds" type="tns:itemQueryByAssetIds" />
<xsd:complexType name="itemQueryByAssetIds">
<xsd:sequence>
<xsd:element minOccurs="0" name="assetIds" nillable="true" type="tns:ArrayOfString" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="itemQueryByAssetIdsResponse"
type="tns:itemQueryByAssetIdsResponse" />
<xsd:complexType name="itemQueryByAssetIdsResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="ns0:ItemComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="deleteMaterial" type="tns:deleteMaterial" />
<xsd:complexType name="deleteMaterial">
<xsd:sequence>
<xsd:element minOccurs="0" name="materialNumber" nillable="true" type="xsd:string" />
</xsd:sequence>
```

```
</xsd:complexType>
<xsd:element name="deleteMaterialResponse" type="tns:deleteMaterialResponse" />
<xsd:complexType name="deleteMaterialResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="ns0:MaterialComposer"
/>
</xsd:sequence>
</xsd:complexType>
<xsd:element name="saveAndUpdateMaterialType"
type="tns:saveAndUpdateMaterialType" />
<xsd:complexType name="saveAndUpdateMaterialType">
<xsd:sequence>
<xsd:element minOccurs="0" name="itemData" nillable="true"
type="ns0:MaterialTypeData" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="saveAndUpdateMaterialTypeResponse"
type="tns:saveAndUpdateMaterialTypeResponse" />
<xsd:complexType name="saveAndUpdateMaterialTypeResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true"
type="ns0:MaterialTypeComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="itemQueryByTrackableKey" type="tns:itemQueryByTrackableKey" />
<xsd:complexType name="itemQueryByTrackableKey">
<xsd:sequence>
<xsd:element minOccurs="0" name="trackableKeys" nillable="true"
type="tns:ArrayOfString" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="itemQueryByTrackableKeyResponse"
type="tns:itemQueryByTrackableKeyResponse" />
<xsd:complexType name="itemQueryByTrackableKeyResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="ns0:ItemComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="itemQueryBySerialNumbers" type="tns:itemQueryBySerialNumbers"
/>
<xsd:complexType name="itemQueryBySerialNumbers">
<xsd:sequence>
<xsd:element minOccurs="0" name="serialNumbers" nillable="true"
type="tns:ArrayOfString" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="itemQueryBySerialNumbersResponse"
type="tns:itemQueryBySerialNumbersResponse" />
<xsd:complexType name="itemQueryBySerialNumbersResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="ns0:ItemComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="itemQueryByTagId" type="tns:itemQueryByTagId" />
<xsd:complexType name="itemQueryByTagId">
<xsd:sequence>
<xsd:element minOccurs="0" name="tagIds" nillable="true" type="tns:ArrayOfString" />
</xsd:sequence>
</xsd:complexType>
```

```
<xsd:element name="itemQueryByTagIdResponse"
type="tns:itemQueryByTagIdResponse" />
<xsd:complexType name="itemQueryByTagIdResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="ns0:ItemComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="getMaterialTypes" type="tns:getMaterialTypes" />
<xsd:complexType name="getMaterialTypes">
<xsd:sequence>
<xsd:element minOccurs="0" name="materialType" nillable="true" type="tns:ArrayOfString"
/>
</xsd:sequence>
</xsd:complexType>
<xsd:element name="getMaterialTypesResponse" type="tns:getMaterialTypesResponse"
/>
<xsd:complexType name="getMaterialTypesResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true"
type="ns0:MaterialTypeComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="saveAndUpdateItem" type="tns:saveAndUpdateItem" />
<xsd:complexType name="saveAndUpdateItem">
<xsd:sequence>
<xsd:element minOccurs="0" name="itemData" nillable="true" type="ns0:ItemData" />
<xsd:element minOccurs="0" name="updateAttribute" nillable="true"
type="ns1:AttributesTypes" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="saveAndUpdateItemResponse"
type="tns:saveAndUpdateItemResponse" />
<xsd:complexType name="saveAndUpdateItemResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="ns0:ItemComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="SOAPException" nillable="true" type="ns2:SOAPException" />
</xsd:schema>
<xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://composer.webservice.context.edgeware.insync.com"
xmlns:tns="http://composer.webservice.context.edgeware.insync.com"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:complexType name="ItemData">
<xsd:sequence>
<xsd:element minOccurs="0" name="alternateTagId" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="appName" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="assetId" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="batchNumber" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="businessLocation" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="containerSerialNumber" nillable="true"
type="xsd:string" />
<xsd:element minOccurs="0" name="containerTrackableNumber" nillable="true"
type="xsd:string" />
<xsd:element minOccurs="0" name="currentLocation" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="currentOwner" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="currentReadPoint" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="currentStorageDetailName" nillable="true"
type="xsd:string" />
```

```
<xsd:element minOccurs="0" name="currentStorageName" nillable="true"
type="xsd:string" />
<xsd:element minOccurs="0" name="currentSubLocation" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="custAttribute1" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute10" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute11" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute12" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute13" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute14" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute15" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute16" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute17" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute18" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute19" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute2" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute20" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute3" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute4" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute5" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute6" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute7" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute8" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute9" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custDateAttribute1" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="custDateAttribute2" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="custDateAttribute3" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="custDateAttribute4" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="custDateAttribute5" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="custNumAttribute1" nillable="true"
type="xsd:decimal" />
<xsd:element minOccurs="0" name="custNumAttribute2" nillable="true"
type="xsd:decimal" />
<xsd:element minOccurs="0" name="custNumAttribute3" nillable="true"
type="xsd:decimal" />
<xsd:element minOccurs="0" name="custNumAttribute4" nillable="true"
type="xsd:decimal" />
<xsd:element minOccurs="0" name="custNumAttribute5" nillable="true"
type="xsd:decimal" />
<xsd:element minOccurs="0" name="customer" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="itemDisposition" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="itemManufacturer" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="itemOwner" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="itemStatus" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="itemType" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="lastLocation" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="lastOperation" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="lastUpdateTime" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="materialNumber" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="netMeasure" nillable="true" type="xsd:double" />
<xsd:element minOccurs="0" name="netMeasureUom" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="netQuantity" nillable="true" type="xsd:double" />
<xsd:element minOccurs="0" name="netQuantityUom" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="nextLocation" nillable="true" type="xsd:string" />
```

```
<xsd:element minOccurs="0" name="operationTimeGMT" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="operationTimeLocal" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="operator" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="ownerLocation" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="partnerReferenceDocumentNumber" nillable="true"
type="xsd:string" />
<xsd:element minOccurs="0" name="referenceDocumentNumber" nillable="true"
type="xsd:string" />
<xsd:element minOccurs="0" name="serialNumber" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="shortDescription" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="supplier" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="systemId" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="tagId" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="timeZone" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="trackableNumber" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="MaterialData">
<xsd:sequence>
<xsd:element minOccurs="0" name="alternateUom" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="batchManagement" type="xsd:boolean" />
<xsd:element minOccurs="0" name="custAttribute1" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute10" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute11" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute12" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute13" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute14" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute15" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute16" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute17" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute18" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute19" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute2" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute20" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute3" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute4" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute5" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute6" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute7" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute8" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute9" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custDateAttribute1" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="custDateAttribute2" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="custDateAttribute3" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="custDateAttribute4" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="custDateAttribute5" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="custNumAttribute1" nillable="true"
type="xsd:decimal" />
<xsd:element minOccurs="0" name="custNumAttribute2" nillable="true"
type="xsd:decimal" />
<xsd:element minOccurs="0" name="custNumAttribute3" nillable="true"
type="xsd:decimal" />
```

```
<xsd:element minOccurs="0" name="custNumAttribute4" nillable="true"
type="xsd:decimal" />
<xsd:element minOccurs="0" name="custNumAttribute5" nillable="true"
type="xsd:decimal" />
<xsd:element minOccurs="0" name="dimensionUom" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="maintenance" type="xsd:boolean" />
<xsd:element minOccurs="0" name="matDepth" nillable="true" type="xsd:double" />
<xsd:element minOccurs="0" name="matHeight" nillable="true" type="xsd:double" />
<xsd:element minOccurs="0" name="matManufacturer" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="matModel" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="matVol" nillable="true" type="xsd:double" />
<xsd:element minOccurs="0" name="matWidth" nillable="true" type="xsd:double" />
<xsd:element minOccurs="0" name="matWt" nillable="true" type="xsd:double" />
<xsd:element minOccurs="0" name="materialBaseUom" nillable="true" type="xsd:string" />
<xsd:element name="MaterialNumber" type="xsd:string" />
<xsd:element minOccurs="0" name="materialType" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="materialVolumeUom" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="materialWeightUom" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="mobileZone" type="xsd:boolean" />
<xsd:element minOccurs="0" name="serialized" type="xsd:boolean" />
<xsd:element minOccurs="0" name="shortDescription" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="status" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="zoneRadius" nillable="true" type="xsd:decimal" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ItemComposer">
<xsd:sequence>
<xsd:element minOccurs="0" name="bizErrorMessage" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="items" nillable="true" type="tns:ArrayOfItemData" />
<xsd:element minOccurs="0" name="organizationName" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="status" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="MaterialComposer">
<xsd:sequence>
<xsd:element minOccurs="0" name="bizErrorMessage" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="materials" nillable="true"
type="tns:ArrayOfMaterialData" />
<xsd:element minOccurs="0" name="organizationName" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="status" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ArrayOfItemData">
<xsd:sequence>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="ItemData" nillable="true"
type="tns:ItemData" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ArrayOfMaterialData">
<xsd:sequence>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="MaterialData"
nillable="true" type="tns:MaterialData" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ArrayOfMaterialTypeData">
```

```
<xsd:sequence>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="MaterialTypeData"
nillable="true" type="tns:MaterialTypeData" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="MaterialTypeData">
<xsd:sequence>
<xsd:element minOccurs="0" name="custAttribute1" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute2" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute3" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute4" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="custAttribute5" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="description" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="materialType" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="trackable" type="xsd:boolean" />
<xsd:element minOccurs="0" name="typeStatus" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="MaterialTypeComposer">
<xsd:sequence>
<xsd:element minOccurs="0" name="bizErrorMessage" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="materialTypes" nillable="true"
type="tns:ArrayOfMaterialTypeData" />
<xsd:element minOccurs="0" name="organizationName" nillable="true" type="xsd:string"
/>
<xsd:element minOccurs="0" name="status" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
  </wsdl:types>
  <wsdl:message name="deleteMaterialResponse">
    <wsdl:part element="tns:deleteMaterialResponse" name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="deleteMaterialType">
    <wsdl:part element="tns:deleteMaterialType" name="parameters">
    </wsdl:part>
    <wsdl:part element="tns:organizationName" name="organizationName">
    </wsdl:part>
    <wsdl:part element="tns:userName" name="userName">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="itemQueryBySystemIds">
    <wsdl:part element="tns:itemQueryBySystemIds" name="parameters">
    </wsdl:part>
    <wsdl:part element="tns:organizationName" name="organizationName">
    </wsdl:part>
    <wsdl:part element="tns:userName" name="userName">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="itemQueryBySerialNumbersResponse">
    <wsdl:part element="tns:itemQueryBySerialNumbersResponse" name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="deleteMaterial">
    <wsdl:part element="tns:deleteMaterial" name="parameters">
    </wsdl:part>
    <wsdl:part element="tns:organizationName" name="organizationName">
    </wsdl:part>
```

```
  <wsdl:part element="tns:userName" name="userName">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="saveAndUpdateMaterialTypeResponse">
  <wsdl:part element="tns:saveAndUpdateMaterialTypeResponse" name="parameters">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="itemQueryBySystemIdsResponse">
  <wsdl:part element="tns:itemQueryBySystemIdsResponse" name="parameters">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="itemQueryByTagIdResponse">
  <wsdl:part element="tns:itemQueryByTagIdResponse" name="parameters">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="itemQueryBySerialNumbers">
  <wsdl:part element="tns:itemQueryBySerialNumbers" name="parameters">
  </wsdl:part>
  <wsdl:part element="tns:organizationName" name="organizationName">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="itemQueryByHistory">
  <wsdl:part element="tns:itemQueryByHistory" name="parameters">
  </wsdl:part>
  <wsdl:part element="tns:organizationName" name="organizationName">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="saveAndUpdateMaterial">
  <wsdl:part element="tns:saveAndUpdateMaterial" name="parameters">
  </wsdl:part>
  <wsdl:part element="tns:organizationName" name="organizationName">
  </wsdl:part>
  <wsdl:part element="tns:userName" name="userName">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="itemQueryByTrackableNumber">
  <wsdl:part element="tns:itemQueryByTrackableNumber" name="parameters">
  </wsdl:part>
  <wsdl:part element="tns:organizationName" name="organizationName">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="itemQueryByTrackableKey">
  <wsdl:part element="tns:itemQueryByTrackableKey" name="parameters">
  </wsdl:part>
  <wsdl:part element="tns:organizationName" name="organizationName">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="getMaterialTypes">
  <wsdl:part element="tns:getMaterialTypes" name="parameters">
  </wsdl:part>
  <wsdl:part element="tns:organizationName" name="organizationName">
  </wsdl:part>
  <wsdl:part element="tns:userName" name="userName">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="itemQueryByAlternateTagIdsResponse">
  <wsdl:part element="tns:itemQueryByAlternateTagIdsResponse" name="parameters">
  </wsdl:part>
</wsdl:message>
```

```
<wsdl:message name="deleteItemByAttribute">
  <wsdl:part element="tns:deleteItemByAttribute" name="parameters">
  </wsdl:part>
  <wsdl:part element="tns:organizationName" name="organizationName">
  </wsdl:part>
  <wsdl:part element="tns:userName" name="userName">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="itemQueryByHistoryResponse">
  <wsdl:part element="tns:itemQueryByHistoryResponse" name="parameters">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="itemQueryByAssetIds">
  <wsdl:part element="tns:itemQueryByAssetIds" name="parameters">
  </wsdl:part>
  <wsdl:part element="tns:organizationName" name="organizationName">
  </wsdl:part>
  <wsdl:part element="tns:userName" name="userName">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="itemQueryByTagId">
  <wsdl:part element="tns:itemQueryByTagId" name="parameters">
  </wsdl:part>
  <wsdl:part element="tns:organizationName" name="organizationName">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="saveAndUpdateItemResponse">
  <wsdl:part element="tns:saveAndUpdateItemResponse" name="parameters">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="saveAndUpdateItem">
  <wsdl:part element="tns:saveAndUpdateItem" name="parameters">
  </wsdl:part>
  <wsdl:part element="tns:organizationName" name="organizationName">
  </wsdl:part>
  <wsdl:part element="tns:userName" name="userName">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="getMaterialsByNumberResponse">
  <wsdl:part element="tns:getMaterialsByNumberResponse" name="parameters">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="itemQueryByTrackableKeyResponse">
  <wsdl:part element="tns:itemQueryByTrackableKeyResponse" name="parameters">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="getMaterialsByNumber">
  <wsdl:part element="tns:getMaterialsByNumber" name="parameters">
  </wsdl:part>
  <wsdl:part element="tns:organizationName" name="organizationName">
  </wsdl:part>
  <wsdl:part element="tns:userName" name="userName">
  </wsdl:part>
</wsdl:message>
<wsdl:message name="saveAndUpdateMaterialType">
  <wsdl:part element="tns:saveAndUpdateMaterialType" name="parameters">
  </wsdl:part>
  <wsdl:part element="tns:organizationName" name="organizationName">
  </wsdl:part>
```

```
    <wsdl:part element="tns:userName" name="userName">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="itemQueryByAlternateTagIds">
    <wsdl:part element="tns:itemQueryByAlternateTagIds" name="parameters">
    </wsdl:part>
    <wsdl:part element="tns:organizationName" name="organizationName">
    </wsdl:part>
    <wsdl:part element="tns:userName" name="userName">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="deleteItemByAttributeResponse">
    <wsdl:part element="tns:deleteItemByAttributeResponse" name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="SOAPException">
    <wsdl:part element="tns:SOAPException" name="SOAPException">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="saveAndUpdateMaterialResponse">
    <wsdl:part element="tns:saveAndUpdateMaterialResponse" name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getMaterialTypesResponse">
    <wsdl:part element="tns:getMaterialTypesResponse" name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="itemQueryByAssetIdsResponse">
    <wsdl:part element="tns:itemQueryByAssetIdsResponse" name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="itemQueryByTrackableNumberResponse">
    <wsdl:part element="tns:itemQueryByTrackableNumberResponse"
name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="deleteMaterialTypeResponse">
    <wsdl:part element="tns:deleteMaterialTypeResponse" name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:portType name="ContextItemService">
    <wsdl:operation name="saveAndUpdateMaterial">
      <wsdl:input message="tns:saveAndUpdateMaterial" name="saveAndUpdateMaterial">
    </wsdl:input>
      <wsdl:output message="tns:saveAndUpdateMaterialResponse"
name="saveAndUpdateMaterialResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="deleteItemByAttribute">
      <wsdl:input message="tns:deleteItemByAttribute" name="deleteItemByAttribute">
    </wsdl:input>
      <wsdl:output message="tns:deleteItemByAttributeResponse"
name="deleteItemByAttributeResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
```

```
    <wsdl:operation name="itemQueryByAlternateTagIds">
      <wsdl:input message="tns:itemQueryByAlternateTagIds"
name="itemQueryByAlternateTagIds">
    </wsdl:input>
      <wsdl:output message="tns:itemQueryByAlternateTagIdsResponse"
name="itemQueryByAlternateTagIdsResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="getMaterialsByNumber">
      <wsdl:input message="tns:getMaterialsByNumber" name="getMaterialsByNumber">
    </wsdl:input>
      <wsdl:output message="tns:getMaterialsByNumberResponse"
name="getMaterialsByNumberResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="itemQueryBySystemIds">
      <wsdl:input message="tns:itemQueryBySystemIds" name="itemQueryBySystemIds">
    </wsdl:input>
      <wsdl:output message="tns:itemQueryBySystemIdsResponse"
name="itemQueryBySystemIdsResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="itemQueryByHistory">
      <wsdl:input message="tns:itemQueryByHistory" name="itemQueryByHistory">
    </wsdl:input>
      <wsdl:output message="tns:itemQueryByHistoryResponse"
name="itemQueryByHistoryResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="deleteMaterialType">
      <wsdl:input message="tns:deleteMaterialType" name="deleteMaterialType">
    </wsdl:input>
      <wsdl:output message="tns:deleteMaterialTypeResponse"
name="deleteMaterialTypeResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="itemQueryByTrackableNumber">
      <wsdl:input message="tns:itemQueryByTrackableNumber"
name="itemQueryByTrackableNumber">
    </wsdl:input>
      <wsdl:output message="tns:itemQueryByTrackableNumberResponse"
name="itemQueryByTrackableNumberResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="itemQueryByAssetIds">
      <wsdl:input message="tns:itemQueryByAssetIds" name="itemQueryByAssetIds">
    </wsdl:input>
```

```
      <wsdl:output message="tns:itemQueryByAssetIdsResponse"
name="itemQueryByAssetIdsResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="deleteMaterial">
      <wsdl:input message="tns:deleteMaterial" name="deleteMaterial">
    </wsdl:input>
      <wsdl:output message="tns:deleteMaterialResponse"
name="deleteMaterialResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="saveAndUpdateMaterialType">
      <wsdl:input message="tns:saveAndUpdateMaterialType"
name="saveAndUpdateMaterialType">
    </wsdl:input>
      <wsdl:output message="tns:saveAndUpdateMaterialTypeResponse"
name="saveAndUpdateMaterialTypeResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="itemQueryByTrackableKey">
      <wsdl:input message="tns:itemQueryByTrackableKey"
name="itemQueryByTrackableKey">
    </wsdl:input>
      <wsdl:output message="tns:itemQueryByTrackableKeyResponse"
name="itemQueryByTrackableKeyResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="itemQueryBySerialNumbers">
      <wsdl:input message="tns:itemQueryBySerialNumbers"
name="itemQueryBySerialNumbers">
    </wsdl:input>
      <wsdl:output message="tns:itemQueryBySerialNumbersResponse"
name="itemQueryBySerialNumbersResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="itemQueryByTagId">
      <wsdl:input message="tns:itemQueryByTagId" name="itemQueryByTagId">
    </wsdl:input>
      <wsdl:output message="tns:itemQueryByTagIdResponse"
name="itemQueryByTagIdResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="getMaterialTypes">
      <wsdl:input message="tns:getMaterialTypes" name="getMaterialTypes">
    </wsdl:input>
      <wsdl:output message="tns:getMaterialTypesResponse"
name="getMaterialTypesResponse">
```

```
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="saveAndUpdateItem">
      <wsdl:input message="tns:saveAndUpdateItem" name="saveAndUpdateItem">
    </wsdl:input>
      <wsdl:output message="tns:saveAndUpdateItemResponse"
name="saveAndUpdateItemResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="ContextItemServiceWSServiceSoapBinding"
type="tns:ContextItemService">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="saveAndUpdateMaterial">
      <soap:operation soapAction="" style="document" />
      <wsdl:input name="saveAndUpdateMaterial">
        <soap:header message="tns:saveAndUpdateMaterial" part="organizationName"
use="literal">
        </soap:header>
        <soap:header message="tns:saveAndUpdateMaterial" part="userName"
use="literal">
        </soap:header>
        <soap:body parts="parameters" use="literal" />
      </wsdl:input>
      <wsdl:output name="saveAndUpdateMaterialResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="SOAPException">
        <soap:fault name="SOAPException" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="deleteItemByAttribute">
      <soap:operation soapAction="" style="document" />
      <wsdl:input name="deleteItemByAttribute">
        <soap:header message="tns:deleteItemByAttribute" part="organizationName"
use="literal">
        </soap:header>
        <soap:header message="tns:deleteItemByAttribute" part="userName" use="literal">
        </soap:header>
        <soap:body parts="parameters" use="literal" />
      </wsdl:input>
      <wsdl:output name="deleteItemByAttributeResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="SOAPException">
        <soap:fault name="SOAPException" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="itemQueryByAlternateTagIds">
      <soap:operation soapAction="" style="document" />
      <wsdl:input name="itemQueryByAlternateTagIds">
        <soap:header message="tns:itemQueryByAlternateTagIds" part="organizationName"
use="literal">
        </soap:header>
        <soap:header message="tns:itemQueryByAlternateTagIds" part="userName"
```

```
use="literal">
      </soap:header>
      <soap:body parts="parameters" use="literal" />
    </wsdl:input>
    <wsdl:output name="itemQueryByAlternateTagIdsResponse">
      <soap:body use="literal" />
    </wsdl:output>
    <wsdl:fault name="SOAPException">
      <soap:fault name="SOAPException" use="literal" />
    </wsdl:fault>
  </wsdl:operation>
  <wsdl:operation name="getMaterialsByNumber">
    <soap:operation soapAction="" style="document" />
    <wsdl:input name="getMaterialsByNumber">
      <soap:header message="tns:getMaterialsByNumber" part="organizationName"
use="literal">
      </soap:header>
      <soap:header message="tns:getMaterialsByNumber" part="userName" use="literal">
      </soap:header>
      <soap:body parts="parameters" use="literal" />
    </wsdl:input>
    <wsdl:output name="getMaterialsByNumberResponse">
      <soap:body use="literal" />
    </wsdl:output>
    <wsdl:fault name="SOAPException">
      <soap:fault name="SOAPException" use="literal" />
    </wsdl:fault>
  </wsdl:operation>
  <wsdl:operation name="itemQueryBySystemIds">
    <soap:operation soapAction="" style="document" />
    <wsdl:input name="itemQueryBySystemIds">
      <soap:header message="tns:itemQueryBySystemIds" part="organizationName"
use="literal">
      </soap:header>
      <soap:header message="tns:itemQueryBySystemIds" part="userName" use="literal">
      </soap:header>
      <soap:body parts="parameters" use="literal" />
    </wsdl:input>
    <wsdl:output name="itemQueryBySystemIdsResponse">
      <soap:body use="literal" />
    </wsdl:output>
    <wsdl:fault name="SOAPException">
      <soap:fault name="SOAPException" use="literal" />
    </wsdl:fault>
  </wsdl:operation>
  <wsdl:operation name="itemQueryByHistory">
    <soap:operation soapAction="" style="document" />
    <wsdl:input name="itemQueryByHistory">
      <soap:header message="tns:itemQueryByHistory" part="organizationName"
use="literal">
      </soap:header>
      <soap:body parts="parameters" use="literal" />
    </wsdl:input>
    <wsdl:output name="itemQueryByHistoryResponse">
      <soap:body use="literal" />
    </wsdl:output>
    <wsdl:fault name="SOAPException">
      <soap:fault name="SOAPException" use="literal" />
    </wsdl:fault>
```

```
    </wsdl:operation>
    <wsdl:operation name="deleteMaterialType">
      <soap:operation soapAction="" style="document" />
      <wsdl:input name="deleteMaterialType">
        <soap:header message="tns:deleteMaterialType" part="organizationName"
use="literal">
        </soap:header>
        <soap:header message="tns:deleteMaterialType" part="userName" use="literal">
        </soap:header>
        <soap:body parts="parameters" use="literal" />
      </wsdl:input>
      <wsdl:output name="deleteMaterialTypeResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="SOAPException">
        <soap:fault name="SOAPException" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="itemQueryByTrackableNumber">
      <soap:operation soapAction="" style="document" />
      <wsdl:input name="itemQueryByTrackableNumber">
        <soap:header message="tns:itemQueryByTrackableNumber"
part="organizationName" use="literal">
        </soap:header>
        <soap:body parts="parameters" use="literal" />
      </wsdl:input>
      <wsdl:output name="itemQueryByTrackableNumberResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="SOAPException">
        <soap:fault name="SOAPException" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="itemQueryByAssetIds">
      <soap:operation soapAction="" style="document" />
      <wsdl:input name="itemQueryByAssetIds">
        <soap:header message="tns:itemQueryByAssetIds" part="organizationName"
use="literal">
        </soap:header>
        <soap:header message="tns:itemQueryByAssetIds" part="userName" use="literal">
        </soap:header>
        <soap:body parts="parameters" use="literal" />
      </wsdl:input>
      <wsdl:output name="itemQueryByAssetIdsResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="SOAPException">
        <soap:fault name="SOAPException" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="deleteMaterial">
      <soap:operation soapAction="" style="document" />
      <wsdl:input name="deleteMaterial">
        <soap:header message="tns:deleteMaterial" part="organizationName" use="literal">
        </soap:header>
        <soap:header message="tns:deleteMaterial" part="userName" use="literal">
        </soap:header>
        <soap:body parts="parameters" use="literal" />
      </wsdl:input>
```

```
      <wsdl:output name="deleteMaterialResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="SOAPException">
        <soap:fault name="SOAPException" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="itemQueryByTrackableKey">
      <soap:operation soapAction="" style="document" />
      <wsdl:input name="itemQueryByTrackableKey">
        <soap:header message="tns:itemQueryByTrackableKey" part="organizationName"
use="literal">
        </soap:header>
        <soap:body parts="parameters" use="literal" />
      </wsdl:input>
      <wsdl:output name="itemQueryByTrackableKeyResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="SOAPException">
        <soap:fault name="SOAPException" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="saveAndUpdateMaterialType">
      <soap:operation soapAction="" style="document" />
      <wsdl:input name="saveAndUpdateMaterialType">
        <soap:header message="tns:saveAndUpdateMaterialType" part="organizationName"
use="literal">
        </soap:header>
        <soap:header message="tns:saveAndUpdateMaterialType" part="userName"
use="literal">
        </soap:header>
        <soap:body parts="parameters" use="literal" />
      </wsdl:input>
      <wsdl:output name="saveAndUpdateMaterialTypeResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="SOAPException">
        <soap:fault name="SOAPException" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="itemQueryBySerialNumbers">
      <soap:operation soapAction="" style="document" />
      <wsdl:input name="itemQueryBySerialNumbers">
        <soap:header message="tns:itemQueryBySerialNumbers" part="organizationName"
use="literal">
        </soap:header>
        <soap:body parts="parameters" use="literal" />
      </wsdl:input>
      <wsdl:output name="itemQueryBySerialNumbersResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="SOAPException">
        <soap:fault name="SOAPException" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="getMaterialTypes">
      <soap:operation soapAction="" style="document" />
      <wsdl:input name="getMaterialTypes">
        <soap:header message="tns:getMaterialTypes" part="organizationName"
```

```
use="literal">
       </soap:header>
       <soap:header message="tns:getMaterialTypes" part="userName" use="literal">
       </soap:header>
       <soap:body parts="parameters" use="literal" />
     </wsdl:input>
     <wsdl:output name="getMaterialTypesResponse">
       <soap:body use="literal" />
     </wsdl:output>
     <wsdl:fault name="SOAPException">
       <soap:fault name="SOAPException" use="literal" />
     </wsdl:fault>
   </wsdl:operation>
   <wsdl:operation name="itemQueryByTagId">
     <soap:operation soapAction="" style="document" />
     <wsdl:input name="itemQueryByTagId">
       <soap:header message="tns:itemQueryByTagId" part="organizationName"
use="literal">
       </soap:header>
       <soap:body parts="parameters" use="literal" />
     </wsdl:input>
     <wsdl:output name="itemQueryByTagIdResponse">
       <soap:body use="literal" />
     </wsdl:output>
     <wsdl:fault name="SOAPException">
       <soap:fault name="SOAPException" use="literal" />
     </wsdl:fault>
   </wsdl:operation>
   <wsdl:operation name="saveAndUpdateItem">
     <soap:operation soapAction="" style="document" />
     <wsdl:input name="saveAndUpdateItem">
       <soap:header message="tns:saveAndUpdateItem" part="organizationName"
use="literal">
       </soap:header>
       <soap:header message="tns:saveAndUpdateItem" part="userName" use="literal">
       </soap:header>
       <soap:body parts="parameters" use="literal" />
     </wsdl:input>
     <wsdl:output name="saveAndUpdateItemResponse">
       <soap:body use="literal" />
     </wsdl:output>
     <wsdl:fault name="SOAPException">
       <soap:fault name="SOAPException" use="literal" />
     </wsdl:fault>
   </wsdl:operation>
 </wsdl:binding>
 <wsdl:service name="ContextItemServiceWSService">
   <wsdl:port binding="tns:ContextItemServiceWSServiceSoapBinding"
name="ContextItemServiceWSPort">
     <soap:address location="http://localhost:8080/sensornet/services/swItemService" />
   </wsdl:port>
 </wsdl:service>
</wsdl:definitions>
```

Service Name: ContextItemService

Methods:

- saveAndUpdateMaterial

  Creates / updates material for the given Enterprise organization

  Note: * fields are mandatory fields

Input Parameter	Description
alternateUom	Alternate Unit of measurement for the material
batchManagement	Represents if a material is maintained in a batch. This identifier needs to be marked if materials are handheld in batches. Mark this indicator as false if batch management is not to be used.
dimensionUom	Unit of measure used for dimensions
Maintenance	Represents if this material is relevant for maintenance. Material marked as true will show up in the maintenance module.
matDepth	Dimensions of the material used for packaging
matHeight	Dimensions of the material used for packaging
matManufacturer	Manufacturer of the material
matModel	What is the part number or the model number of the manufacturer? Or how the manufacturer does identifies this material in his own system. This is also used to identify the material in manufacturer's system coz our identification of material using material # may not be the same way manufacturer uses their model #.
matVo	Represents the volume of the material
matWidth	Dimensions of the material used for packaging
matWt	Represents the weight of the material
materialBaseUom *	Base Unit of measurement for the material
materialNumber *	This number uniquely identifies a material
materialType	Material type of the material
materialVolume	Represents the volume of the material
materialWeightUom	Unit of measure for weight
mobileZone *	Insync special: if we want to do some kind of alerts in RTLS, then one needs to set this as true. 1= True, 0= false. Ex: Alarms can be raised if a person is plotted near the storage area of hazardous materials, which may be a no go zone for an individual without proper cover.
Serialized	Helps identifies if a material is to be tracked at the individual level or not. This is set to true in case we want to track individually
shortDescription	Brief description of the material
Status *	ACTIVE/INACTIVE : Flag will represent whether the material is active/inactive in edgeware for the given organization
zoneRadius	Radius can be set for mobile_zone which defines the area which is marked as no go zone.
custAttribute1 – custAttribute20	Additional String attribute provided ,used to store any application/Biz data
custNumAttribute1-custNumAttribute5	Additional Number attribute provided ,used to store any application/Biz data

- deleteItemByAttribute

  Deletes Material for the given material number and organization

Input Parameter	Description
materialNumber *	This number uniquely identifies a material

- saveAndUpdateMaterialType

  Creates / updates material type for the given enterprise organization

Input_Parameter	Description
MaterialType	Material Type
trackable	
Status *	ACTIVE/INACTIVE : Flag will represent whether the material type is active/inactive in edgeware for the given organization
custAttribute1 – custAttribute5	Additional String attribute provided ,used to store any application/Biz data

- deleteMaterialType

Input_Parameter	Description
MaterialType	Material type

- saveAndUpdateItem

  Creates/Updates a trackable item in edgeware for the given organization and type of tagid.

Input Parameter	Description
alternateTagId	Unique tagid for a given item
appName *	App name configured in edgeware
assetId	Unique tagid for a given item
systemId	Unique tagid for a given item
batchNumber	
businessLocation	Biz Location of an item that is configured in edgeware
containerSerialNumber	Represents serial number of the container (Parent) this item belong to
containerTrackableNumber	Represents Trackable number of the container (Parent) this item belong to
currentLocation	Location at which item is currently available/present.
currentOwner	Currently who owns the item [
currentReadPoint	From which read point a read is received
currentStorageDetailName	Represents the 2nd level Storage where the item is located.
currentStorageName	Represents the 1st level Storage where the item is located.
currentSubLocation	Grid / Region in which this item is currently located.
customer	Customer .This customer should be configured in edgeware
itemDisposition	Gives the state of the item. Ex: new, available, repair,

	issued, picking, etc…
itemManufacturer	Manufacturer of the Item
itemOwner	Original owner of the Item
itemStatus	It can have values of active and inactive only. At times, one more status of unavailable can also be allowed. Primarily used to identify if this item is part of the process or not.
lastLocation	Stores the previous location of the item
lastOperation	Stores the last operation defined on the item
lastUpdateTime	GMT time at which the item is persisted
materialNumber	Material Number
netMeasure	Volume of units in this Asset
netMeasureUom	Unit of measure for net measure
netQuantity	Number of units in the Asset.
netQuantityUom	Unit of measure for net quantity
nextLocation	The next location the item, it is expected to move.
operationTimeGMT	GMT time at which the item is persisted
operator	Who performed this operation? This can be found from the login credentials used.
ownerLocation	At times items may be recognized/belongs/owned by/with locations. This fields helps to find the owner location.
partnerReferenceDocumentNumber	Insync reference document. Document with which an item is either bought or sold.
referenceDocumentNumber	Partner document number. Ex: invoice number, partner delivery number, etc…
serialNumber	This is a manufacturer serial number.
shortDescription	Brief description of the item
supplier	Represents the supplier for the given organization. This should be a valid supplier configured in edgeware
tagId	Id of the RFID tag attached to the Asset which has to be unique. Same tag id serial number cannot be given to multiple Assets. If tags are to be reused, it should be disassociated with the previous Asset id. Shouldn't be used for historical reporting.
trackableKey	Edgeware generated number for uniquely maintaining a record in database.
trackableNumber *	This number has to be a unique number that is either generated by number or imported by the application. This field can be filled with serial number generated by the company and not by the manufacturer. Historical reports should be based only on trackable number.
timeZone	Time zone obtained by the current location
custAttribute1 – custAttribute20	Additional String attribute provided ,used to store any application / Biz data
custNumAttribute1-custNumAttribute5	Additional Number attribute provided ,used to store any application/Biz data
custDateAttribute1 - custDateAttribute5	Additional Date attribute provided ,used to store any application/Biz data
updateAttribute	This should have values : CREATE_ITEM,TRACKABLE_NUMBER,TAG_ID,SERIAL_NUMBER,ASSET_ID,SYSTEM_ID,ALTERNATE_TAG_ID

	CREATE_ITEM : Creating item for the first time. TRACKABLE_NUMBER, TAG_ID, SERIAL_NUMBER, ASSET_ID, SYSTEM_ID, and ALTERNATE_TAG_ID: Update an item .The item will be fetched and updated for the given property.

- deleteItemByAttribute

  Delete a trackable item in edgeware for the given organization and type of update attribute passed.

  For example if attribute type is ALTERNATE_TAG_ID and attribute value is AT1001.Then the system will fetch the item for ALTERNATE_TAG_ID = AT1001. If found it deletes the item from the system.

Input Parameter	Description
attributeType	Possible values are TRACKABLE_NUMBER, TAG_ID, SERIAL_NUMBER, ASSET_ID, SYSTEM_ID, and ALTERNATE_TAG_ID
attributeValue	This can have the unique value representing the item

- itemQueryByAlternateTagIds

  Retrieve item data for list of alternate tag ids provided.

Input Parameter	Description
alternateTagIds	One or more alternate tag ids

- getMaterialsByNumber

  Retrieve material data for one or more material numbers

Input Parameter	Description
MaterialNumbers	One or more material number

- itemQueryBySystemIds

  Retrieve item data for list of system ids provided.

Input Parameter	Description
systemIds	One or more system ids

- itemQueryByAssetIds

  Retrieve item data for list of asset ids provided.

Input Parameter	Description
AssetIds	One or more Asset ids

- itemQueryByTagId

  Retrieve item data for list of tag ids provided.

Input Parameter	Description
TagIds	One or more tag ids

- itemQueryBySerialNumbers

  Retrieve item data for list of serial numbers provided.

Input Parameter	Description
SerialNumbers	One or more serial numbers

- itemQueryByHistory

  Fetch history of the tag between two operation dates and for a given tag id

Input Parameter	Description
TagId	Tag Id of the item
startDate	Operation start time in gmt when the tag might have read
endDate	Operation start time in gmt when the tag might have read

- itemQueryByTrackableNumber

  Retrieve item data for list of trackable numbers provided.

Input Parameter	Description
trackableNumbers	One or more trackable numbers

- deleteMaterial

  Deletes material in edgeware for the given material number and enterprise organization.

Input Parameter	Description
materialNumber *	This number uniquely identifies a material

- getMaterialTypes

  Retrieve material type data for one or more material types.

Input Parameter	Description
MaterialTypes	One or more material types

19.4 Posting Tag Read and RTLS Tag read event

To view the wsdl file access the URL after running sensornet.

http://<ipaddress>:<portnumber>/sensornet/services/tagReadWebservice?wsdl

where
<ipaddress> is the hostname/ipaddress of the server running sensornet
<portnumber> is the port on which tomcat is running.

WSDL:

```
<?xml version='1.0' encoding='UTF-8'?><wsdl:definitions
name="TagReadWebserviceWSService"
targetNamespace="http://service.webservice.sensornet.edgeware.insync.com/"
xmlns:ns1="http://cxf.apache.org/bindings/xformat"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://service.webservice.sensornet.edgeware.insync.com/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:types>
<xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://lang.java" xmlns:tns="http://lang.java"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:complexType name="Throwable">
<xsd:sequence />
</xsd:complexType>
</xsd:schema>
<xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://soap.xml.javax" xmlns:ns0="http://lang.java"
xmlns:tns="http://soap.xml.javax" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:import namespace="http://lang.java" />
<xsd:complexType name="SOAPException">
<xsd:sequence>
<xsd:element minOccurs="0" name="cause" nillable="true" type="ns0:Throwable" />
<xsd:element minOccurs="0" name="message" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
<xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://composer.webservice.sensornet.edgeware.insync.com"
xmlns:tns="http://composer.webservice.sensornet.edgeware.insync.com"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:complexType name="TagDataComposer">
<xsd:sequence>
<xsd:element minOccurs="0" name="XLon" type="xsd:double" />
<xsd:element minOccurs="0" name="YLat" type="xsd:double" />
<xsd:element minOccurs="0" name="ZAlt" type="xsd:double" />
<xsd:element minOccurs="0" name="Antenna" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="parameters" nillable="true"
type="tns:ArrayOfParameterComposer" />
<xsd:element name="TagID" nillable="true" type="xsd:string" />
<xsd:element name="TRTime" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ArrayOfEventComposer">
<xsd:sequence>
```

```
<xsd:element maxOccurs="unbounded" minOccurs="0" name="EventComposer"
nillable="true" type="tns:EventComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ArrayOfRtlsTagDataComposer">
<xsd:sequence>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="RtlsTagDataComposer"
nillable="true" type="tns:RtlsTagDataComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ArrayOfTagDataComposer">
<xsd:sequence>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="TagDataComposer"
nillable="true" type="tns:TagDataComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ParameterComposer">
<xsd:sequence>
<xsd:element minOccurs="0" name="name" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="type" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="value" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="EventComposer">
<xsd:sequence>
<xsd:element minOccurs="0" name="Device" type="xsd:string" />
<xsd:element minOccurs="0" name="SensornetEventName" nillable="true"
type="xsd:string" />
<xsd:element name="Location" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="ExtraParameters"
type="tns:ArrayOfParameterComposer" />
<xsd:element minOccurs="0" name="RequestId" nillable="true" type="xsd:string" />
<xsd:element name="TagData" nillable="true" type="tns:ArrayOfTagDataComposer" />
<xsd:element minOccurs="0" name="Time" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="RtlsTagDataComposer">
<xsd:sequence>
<xsd:element minOccurs="0" name="XLon" type="xsd:double" />
<xsd:element minOccurs="0" name="YLat" type="xsd:double" />
<xsd:element minOccurs="0" name="ZAlt" type="xsd:double" />
<xsd:element minOccurs="0" name="Accuracy" type="xsd:string" />
<xsd:element minOccurs="0" name="Antenna" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="BatteryStatus" type="xsd:string" />
<xsd:element minOccurs="0" name="BatteryVoltage" type="xsd:string" />
<xsd:element minOccurs="0" name="Direction" type="xsd:string" />
<xsd:element minOccurs="0" name="ExtraParameters"
type="tns:ArrayOfParameterComposer" />
<xsd:element minOccurs="0" name="Region" type="xsd:string" />
<xsd:element name="TagID" type="xsd:string" />
<xsd:element minOccurs="0" name="TRTime" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ArrayOfRtlsEventComposer">
<xsd:sequence>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="RtlsEventComposer"
nillable="true" type="tns:RtlsEventComposer" />
</xsd:sequence>
</xsd:complexType>
```

```
<xsd:complexType name="ArrayOfParameterComposer">
<xsd:sequence>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="ParameterComposer"
nillable="true" type="tns:ParameterComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="RtlsEventComposer">
<xsd:sequence>
<xsd:element name="SensornetEventName" nillable="true" type="xsd:string" />
<xsd:element name="Location" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="ExtraParameters"
type="tns:ArrayOfParameterComposer" />
<xsd:element minOccurs="0" name="PostionServer" nillable="true" type="xsd:string" />
<xsd:element minOccurs="0" name="RequestId" nillable="true" type="xsd:string" />
<xsd:element name="TagData" nillable="true" type="tns:ArrayOfRtlsTagDataComposer"
/>
<xsd:element minOccurs="0" name="Time" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
<xsd:schema attributeFormDefault="unqualified" elementFormDefault="unqualified"
targetNamespace="http://service.webservice.sensornet.edgeware.insync.com/"
xmlns:ns0="http://composer.webservice.sensornet.edgeware.insync.com"
xmlns:ns1="http://soap.xml.javax"
xmlns:tns="http://service.webservice.sensornet.edgeware.insync.com/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:import namespace="http://composer.webservice.sensornet.edgeware.insync.com"
/>
<xsd:element name="postRtlsTagReads" type="tns:postRtlsTagReads" />
<xsd:complexType name="postRtlsTagReads">
<xsd:sequence>
<xsd:element minOccurs="0" name="events" nillable="true"
type="ns0:ArrayOfRtlsEventComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="organizationName" type="xsd:string" />
<xsd:element name="userName" type="xsd:string" />
<xsd:element name="postRtlsTagReadsResponse"
type="tns:postRtlsTagReadsResponse" />
<xsd:complexType name="postRtlsTagReadsResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="postTagReads" type="tns:postTagReads" />
<xsd:complexType name="postTagReads">
<xsd:sequence>
<xsd:element minOccurs="0" name="events" nillable="true"
type="ns0:ArrayOfEventComposer" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="postTagReadsResponse" type="tns:postTagReadsResponse" />
<xsd:complexType name="postTagReadsResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="SOAPException" nillable="true" type="ns1:SOAPException" />
</xsd:schema>
```

```
  </wsdl:types>
  <wsdl:message name="postRtlsTagReads">
    <wsdl:part element="tns:postRtlsTagReads" name="parameters">
    </wsdl:part>
    <wsdl:part element="tns:organizationName" name="organizationName">
    </wsdl:part>
    <wsdl:part element="tns:userName" name="userName">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="SOAPException">
    <wsdl:part element="tns:SOAPException" name="SOAPException">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="postRtlsTagReadsResponse">
    <wsdl:part element="tns:postRtlsTagReadsResponse" name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="postTagReads">
    <wsdl:part element="tns:postTagReads" name="parameters">
    </wsdl:part>
    <wsdl:part element="tns:organizationName" name="organizationName">
    </wsdl:part>
    <wsdl:part element="tns:userName" name="userName">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="postTagReadsResponse">
    <wsdl:part element="tns:postTagReadsResponse" name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:portType name="TagReadWebservice">
    <wsdl:operation name="postRtlsTagReads">
      <wsdl:input message="tns:postRtlsTagReads" name="postRtlsTagReads">
    </wsdl:input>
      <wsdl:output message="tns:postRtlsTagReadsResponse"
name="postRtlsTagReadsResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="postTagReads">
      <wsdl:input message="tns:postTagReads" name="postTagReads">
    </wsdl:input>
      <wsdl:output message="tns:postTagReadsResponse"
name="postTagReadsResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="TagReadWebserviceWSServiceSoapBinding"
type="tns:TagReadWebservice">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="postRtlsTagReads">
      <soap:operation soapAction="" style="document" />
      <wsdl:input name="postRtlsTagReads">
        <soap:header message="tns:postRtlsTagReads" part="organizationName"
use="literal">
        </soap:header>
        <soap:header message="tns:postRtlsTagReads" part="userName" use="literal">
```

```
        </soap:header>
        <soap:body parts="parameters" use="literal" />
      </wsdl:input>
      <wsdl:output name="postRtlsTagReadsResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="SOAPException">
        <soap:fault name="SOAPException" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="postTagReads">
      <soap:operation soapAction="" style="document" />
      <wsdl:input name="postTagReads">
        <soap:header message="tns:postTagReads" part="organizationName"
use="literal">
        </soap:header>
        <soap:header message="tns:postTagReads" part="userName" use="literal">
        </soap:header>
        <soap:body parts="parameters" use="literal" />
      </wsdl:input>
      <wsdl:output name="postTagReadsResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="SOAPException">
        <soap:fault name="SOAPException" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="TagReadWebserviceWSService">
    <wsdl:port binding="tns:TagReadWebserviceWSServiceSoapBinding"
name="TagReadWebserviceWSPort">
      <soap:address
location="http://localhost:8080/sensornet/services/tagReadWebservice" />
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

Service: tagReadWebservice

Methods:

- postTagReads

  This method will post tag read event in edgeware

Input Parameter	Description
SensornetEventName *	Valid tag read event configured in edgeware.
Device	Name of the device through which the tag is read. This device should be a device configured in edgeware.
Location	Name of a location where the tag is read. This location should be configured in our system
TagData. XLon	Position of the tag .In can be a geometric position in a place or the Longitude at that point.
TagData. YLat	Position of the tag .In can be a geometric position in a place or the Latitude at point.
TagData. ZAlt	Position of the tag .In can be a geometric position in a place or the Altitude at that point.

TagData. TagID	Tag id of the item
TagData. TRTime	Time at which the tag was read .It should follow the ISO format [**yyyy-MM-dd'T'HH:mm:ss.SSS'Z'**]
TagData. ParameterComposer	Additional data for the tag can be represented here in name/value pairs

- postRtlsTagReads

  This method will post rtls tag read event in edgeware

Input Parameter	**Description**
SensornetEventName *	Valid Rtls tag read event configured in edgeware.
Location *	Name of a location where the tag is read. This location should be configured in our system
TagData. XLon	Position of the tag .In can be a geometric position in a place or the Longitude at that point.
TagData. YLat	Position of the tag .In can be a geometric position in a place or the Latitude at point.
TagData. ZAlt	Position of the tag .In can be a geometric position in a place or the Altitude at that point.
TagData. TagID	Tag id of the item
TagData. TRTime	Time at which the tag was read .It should follow the ISO format [**yyyy-MM-dd'T'HH:mm:ss.SSS'Z'**]
TagData. ParameterComposer	Additional data for the tag can be represented here in name/value pairs
RequestId	For future release
TagData.Accuracy	Accuracy of the read
TagData.Antenna	The name of the antenna ,the tag was read
TagData.BatteryStatus	Battery status of the active tag
TagData.BatteryVoltage	Battery voltage of the active tag
TagData.Direction	Valid direction of the tag .

To view the wsdl file access the URL after running sensornet.

http://localhost:8080/sensornet/services/plotTagWebservice?wsdl

where
<ipaddress> is the hostname/ipaddress of the server running sensornet
<portnumber> is the port on which tomcat is running

Service: plotTagWebservice

WSDL:

```
<?xml version='1.0' encoding='UTF-8'?><wsdl:definitions
name="PlotTagWebserviceWSService"
targetNamespace="http://service.webservice.sensornet.edgeware.insync.com/"
xmlns:ns1="http://cxf.apache.org/bindings/xformat"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://service.webservice.sensornet.edgeware.insync.com/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:types>
```

```
<xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://service.webservice.sensornet.edgeware.insync.com/"
xmlns:ns0="http://soap.xml.javax"
xmlns:tns="http://service.webservice.sensornet.edgeware.insync.com/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:complexType name="ArrayOfString">
<xsd:sequence>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="string" nillable="true"
type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="getPlotTagUrl" type="tns:getPlotTagUrl" />
<xsd:complexType name="getPlotTagUrl">
<xsd:sequence>
<xsd:element minOccurs="0" name="tagIds" nillable="true" type="tns:ArrayOfString" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="organizationName" type="xsd:string" />
<xsd:element name="userName" type="xsd:string" />
<xsd:element name="serverIpAddress" type="xsd:string" />
<xsd:element name="serverPortNumber" type="xsd:string" />
<xsd:element name="contextName" type="xsd:string" />
<xsd:element name="getPlotTagUrlResponse" type="tns:getPlotTagUrlResponse" />
<xsd:complexType name="getPlotTagUrlResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="tns:ArrayOfString" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="SOAPException" nillable="true" type="ns0:SOAPException" />
</xsd:schema>
<xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://lang.java" xmlns:tns="http://lang.java"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:complexType name="Throwable">
<xsd:sequence />
</xsd:complexType>
</xsd:schema>
<xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://soap.xml.javax" xmlns:ns0="http://lang.java"
xmlns:tns="http://soap.xml.javax" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:import namespace="http://lang.java" />
<xsd:complexType name="SOAPException">
<xsd:sequence>
<xsd:element minOccurs="0" name="cause" nillable="true" type="ns0:Throwable" />
<xsd:element minOccurs="0" name="message" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
  </wsdl:types>
  <wsdl:message name="SOAPException">
    <wsdl:part element="tns:SOAPException" name="SOAPException">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getPlotTagUrlResponse">
    <wsdl:part element="tns:getPlotTagUrlResponse" name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getPlotTagUrl">
    <wsdl:part element="tns:getPlotTagUrl" name="parameters">
```

```
    </wsdl:part>
    <wsdl:part element="tns:organizationName" name="organizationName">
    </wsdl:part>
    <wsdl:part element="tns:userName" name="userName">
    </wsdl:part>
    <wsdl:part element="tns:serverIpAddress" name="serverIpAddress">
    </wsdl:part>
    <wsdl:part element="tns:serverPortNumber" name="serverPortNumber">
    </wsdl:part>
    <wsdl:part element="tns:contextName" name="contextName">
    </wsdl:part>
  </wsdl:message>
  <wsdl:portType name="PlotTagWebservice">
    <wsdl:operation name="getPlotTagUrl">
      <wsdl:input message="tns:getPlotTagUrl" name="getPlotTagUrl">
    </wsdl:input>
      <wsdl:output message="tns:getPlotTagUrlResponse"
name="getPlotTagUrlResponse">
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="PlotTagWebserviceWSServiceSoapBinding"
type="tns:PlotTagWebservice">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="getPlotTagUrl">
      <soap:operation soapAction="" style="document" />
      <wsdl:input name="getPlotTagUrl">
        <soap:header message="tns:getPlotTagUrl" part="organizationName"
use="literal">
        </soap:header>
        <soap:header message="tns:getPlotTagUrl" part="userName" use="literal">
        </soap:header>
        <soap:header message="tns:getPlotTagUrl" part="serverIpAddress" use="literal">
        </soap:header>
        <soap:header message="tns:getPlotTagUrl" part="serverPortNumber"
use="literal">
        </soap:header>
        <soap:header message="tns:getPlotTagUrl" part="contextName" use="literal">
        </soap:header>
        <soap:body parts="parameters" use="literal" />
      </wsdl:input>
      <wsdl:output name="getPlotTagUrlResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="SOAPException">
        <soap:fault name="SOAPException" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="PlotTagWebserviceWSService">
    <wsdl:port binding="tns:PlotTagWebserviceWSServiceSoapBinding"
name="PlotTagWebserviceWSPort">
      <soap:address
location="http://localhost:8080/sensornet/services/plotTagWebservice" />
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

Methods:

- getPlotTagUrl

  This method will fetch URL's .This URL's can be used to view the items on a map configured by edgeware.

  The items to be plotted have to be sent in this method.

Input Parameter	Description
tagIds	Tag Id's of an item that has to be plotted on a map

19.5 URL Generation for tag plotting on image

To view the wsdl file access the URL after running sensornet.

http://localhost:8080/sensornet/services/plotTagWebservice?wsdl

where
<ipaddress> is the hostname/ipaddress of the server running sensornet
<portnumber> is the port on which tomcat is running

Service: plotTagWebservice

WSDL:

```
<?xml version='1.0' encoding='UTF-8'?><wsdl:definitions
name="PlotTagWebserviceWSService"
targetNamespace="http://service.webservice.sensornet.edgeware.insync.com/"
xmlns:ns1="http://cxf.apache.org/bindings/xformat"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://service.webservice.sensornet.edgeware.insync.com/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:types>
<xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://service.webservice.sensornet.edgeware.insync.com/"
xmlns:ns0="http://soap.xml.javax"
xmlns:tns="http://service.webservice.sensornet.edgeware.insync.com/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:complexType name="ArrayOfString">
<xsd:sequence>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="string" nillable="true"
type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="getPlotTagUrl" type="tns:getPlotTagUrl" />
<xsd:complexType name="getPlotTagUrl">
<xsd:sequence>
<xsd:element minOccurs="0" name="tagIds" nillable="true" type="tns:ArrayOfString" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="organizationName" type="xsd:string" />
```

```
<xsd:element name="userName" type="xsd:string" />
<xsd:element name="serverIpAddress" type="xsd:string" />
<xsd:element name="serverPortNumber" type="xsd:string" />
<xsd:element name="contextName" type="xsd:string" />
<xsd:element name="getPlotTagUrlResponse" type="tns:getPlotTagUrlResponse" />
<xsd:complexType name="getPlotTagUrlResponse">
<xsd:sequence>
<xsd:element minOccurs="0" name="return" nillable="true" type="tns:ArrayOfString" />
</xsd:sequence>
</xsd:complexType>
<xsd:element name="SOAPException" nillable="true" type="ns0:SOAPException" />
</xsd:schema>
<xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://lang.java" xmlns:tns="http://lang.java"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:complexType name="Throwable">
<xsd:sequence />
</xsd:complexType>
</xsd:schema>
<xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://soap.xml.javax" xmlns:ns0="http://lang.java"
xmlns:tns="http://soap.xml.javax" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:import namespace="http://lang.java" />
<xsd:complexType name="SOAPException">
<xsd:sequence>
<xsd:element minOccurs="0" name="cause" nillable="true" type="ns0:Throwable" />
<xsd:element minOccurs="0" name="message" nillable="true" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
  </wsdl:types>
  <wsdl:message name="SOAPException">
    <wsdl:part element="tns:SOAPException" name="SOAPException">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getPlotTagUrlResponse">
    <wsdl:part element="tns:getPlotTagUrlResponse" name="parameters">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="getPlotTagUrl">
    <wsdl:part element="tns:getPlotTagUrl" name="parameters">
    </wsdl:part>
    <wsdl:part element="tns:organizationName" name="organizationName">
    </wsdl:part>
    <wsdl:part element="tns:userName" name="userName">
    </wsdl:part>
    <wsdl:part element="tns:serverIpAddress" name="serverIpAddress">
    </wsdl:part>
    <wsdl:part element="tns:serverPortNumber" name="serverPortNumber">
    </wsdl:part>
    <wsdl:part element="tns:contextName" name="contextName">
    </wsdl:part>
  </wsdl:message>
  <wsdl:portType name="PlotTagWebservice">
    <wsdl:operation name="getPlotTagUrl">
      <wsdl:input message="tns:getPlotTagUrl" name="getPlotTagUrl">
    </wsdl:input>
      <wsdl:output message="tns:getPlotTagUrlResponse"
name="getPlotTagUrlResponse">
```

```
    </wsdl:output>
      <wsdl:fault message="tns:SOAPException" name="SOAPException">
    </wsdl:fault>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="PlotTagWebserviceWSServiceSoapBinding"
type="tns:PlotTagWebservice">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="getPlotTagUrl">
      <soap:operation soapAction="" style="document" />
      <wsdl:input name="getPlotTagUrl">
        <soap:header message="tns:getPlotTagUrl" part="organizationName"
use="literal">
        </soap:header>
        <soap:header message="tns:getPlotTagUrl" part="userName" use="literal">
        </soap:header>
        <soap:header message="tns:getPlotTagUrl" part="serverIpAddress" use="literal">
        </soap:header>
        <soap:header message="tns:getPlotTagUrl" part="serverPortNumber"
use="literal">
        </soap:header>
        <soap:header message="tns:getPlotTagUrl" part="contextName" use="literal">
        </soap:header>
        <soap:body parts="parameters" use="literal" />
      </wsdl:input>
      <wsdl:output name="getPlotTagUrlResponse">
        <soap:body use="literal" />
      </wsdl:output>
      <wsdl:fault name="SOAPException">
        <soap:fault name="SOAPException" use="literal" />
      </wsdl:fault>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="PlotTagWebserviceWSService">
    <wsdl:port binding="tns:PlotTagWebserviceWSServiceSoapBinding"
name="PlotTagWebserviceWSPort">
      <soap:address
location="http://localhost:8080/sensornet/services/plotTagWebservice" />
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

Methods:

- getPlotTagUrl

  This method will fetch URL's .This URL's can be used to view the items on a map configured by edgeware.

  The items to be plotted have to be sent in this method.

Input Parameter	**Description**

tagIds	Tag Id’s of an item that has to be plotted on a map